US011070812B2

(12) **United States Patent**
Coban et al.

(10) **Patent No.: US 11,070,812 B2**
(45) **Date of Patent: Jul. 20, 2021**

(54) **COEFFICIENT DOMAIN BLOCK DIFFERENTIAL PULSE-CODE MODULATION IN VIDEO CODING**

(71) Applicant: **QUALCOMM Incorporated**, San Diego, CA (US)

(72) Inventors: **Muhammed Zeyd Coban**, Carlsbad, CA (US); **Marta Karczewicz**, San Diego, CA (US)

(73) Assignee: **QUALCOMM Incorporated**, San Diego, CA (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

(21) Appl. No.: **16/816,116**

(22) Filed: **Mar. 11, 2020**

(65) **Prior Publication Data**

US 2020/0296381 A1 Sep. 17, 2020

Related U.S. Application Data

(60) Provisional application No. 62/817,451, filed on Mar. 12, 2019.

(51) **Int. Cl.**
H04N 19/132 (2014.01)
H04N 19/105 (2014.01)
(Continued)

(52) **U.S. Cl.**
CPC ***H04N 19/132*** (2014.11); ***H04N 19/105*** (2014.11); ***H04N 19/159*** (2014.11); ***H04N 19/176*** (2014.11); ***H04N 19/70*** (2014.11)

(58) **Field of Classification Search**
USPC 375/240.02
See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

2014/0286412 A1* 9/2014 Joshi H04N 19/467 375/240.12
2014/0286413 A1 9/2014 Joshi et al.
(Continued)

FOREIGN PATENT DOCUMENTS

WO 2014160714 10/2014

OTHER PUBLICATIONS

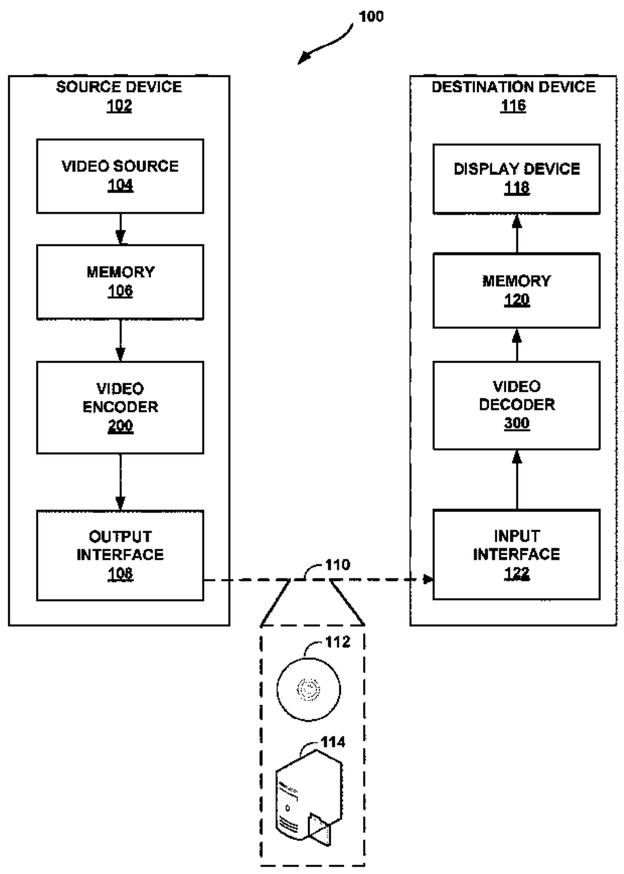
Bossen F., et al., "JVET Common Test Conditions and Software Reference Configurations for SDR video", Joint Video Experts Team (JVET) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11, Document: JVET-M1010-V1, 13th Meeting: Marrakech, MA, Jan. 9-18, 2019, pp. 1-6.
(Continued)

Primary Examiner — Behrooz M Senfi
(74) *Attorney, Agent, or Firm* — Shumaker & Sieffert, P.A.

(57) **ABSTRACT**

A video decoder may determine, based on syntax elements in a bitstream that comprises an encoded representation of the video data, residual quantized samples of a block of the video data. Additionally, the video decoder may determine quantized residual values based on the residual quantized samples. After determining the quantized residual values, the video decoder may inverse quantize the quantized residual values. The video decoder may generate predicted values by performing intra prediction for the block using unfiltered samples from above or left block boundary samples. Furthermore, the video decoder may reconstruct original sample values of the block based on the inverse-quantized quantized residual values and the prediction values.

11 Claims, 8 Drawing Sheets



(51) **Int. Cl.**

H04N 19/176	(2014.01)
H04N 19/159	(2014.01)
H04N 19/70	(2014.01)

(56) **References Cited**

U.S. PATENT DOCUMENTS

2014/0362917	A1*	12/2014	Joshi	H04N 19/593 375/240.13
2020/0275111	A1*	8/2020	Zhao	H04N 19/18
2020/0275121	A1*	8/2020	Zhao	H04N 19/18
2020/0296390	A1*	9/2020	Chao	H04N 19/176

OTHER PUBLICATIONS

Abdoli M., et al., "AHG11: Block DPCM for Screen Content Coding", Document: JVET-L0078, Joint Video Experts Team (JVET) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11, 12th Meeting: Macao, CN, Oct. 3-12, 2018, JVET-L0078-v2, pp. 1-7.

Abdoli M., et al., "CE8: BDPCM with Horizontal/vertical Predictor and Independently Decodable Areas (test 8.3.1b)", Joint Video Experts Team (JVET) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11, 13th Meeting: Marrakech, MA, Jan. 9-18, 2019, Document: JVET-M0057, pp. 1-7.

Bossen F., et al., "JEM Software Manual," Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11, Document: JCTVC-Software Manual, Retrieved on Aug. 3, 2016, pp. 1-29.

Bross B., et al., "Versatile Video Coding (Draft 4)", Joint Video Experts Team (JVET) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11, 13th Meeting: Marrakech, MA, Jan. 9-18, 2019, JVET-M1001-v5, 287 Pages.

Chen J., et al., "Algorithm Description of Joint Exploration Test Model 1," 1, JVET Meeting, Oct. 19-21, 2015, Geneva;(The Joint Video Exploration Team of ISO/IEC JTC1/SC29NVG11 and ITU-T SG.16); URL: http://phenix.int-evry.fr/jvet/ ,, No. JVET-A1001 Feb. 24, 2016 (Feb. 24, 2016), XP030150000, 27 Pages.

International Search Report and Written Opinion—PCT/US2020/022376—ISA/EPO—May 27, 2020, 16 Pages.

ITU-T H.265, "Series H: Audiovisual and Multimedia Systems, Infrastructure of Audiovisual Services—Coding of Moving Video, High efficiency Video Coding," The International Telecommunication Union. Dec. 2016, 664 Pages.

Joshi R., et al., "RCE2 subtest C.2: Extension of residual DPCM to lossy coding", 14. JCT-VC Meeting; Jul. 25, 2013-Feb. 8, 2013; Vienna; (Joint Collaborative Team on Video Coding of ISO/IEC JTC1/SC29/WG11 and ITU-T SG.16); URL: http://wftp3.itu.int/av-arch/jctvc-site/,, No. JCTVC-N0052, Jul. 16, 2013 (Jul. 16, 2013), XP030114481, 3 Pages.

Karczewicz (Qualcomm) M., et al., "CE8-Related: Quantized Residual BDPCM", 14. JVET Meeting, Mar. 19, 2019-Mar. 27, 2019, Geneva, (The Joint Video Exploration Team of ISO/IEC JTC1/SC29/WG11 and ITU-T SG.16) , No. JVET-N0413, May 14, 2019 (May 14, 2019), XP030205192, 5 Pages, Retrieved from the Internet: URL: http://phenix.int-evry.fr/jvet/doc_end_user/documents/14_Geneva/wg11/JVET-N0413-v6.zip JVET-N0413 r3.docx. [retrieved on May 14, 2019] the whole document.

Naccari M., et al., "On Residual DPCM Design Unification for Lossy Coding in HEVC-RExt", 15. JCT-VC Meeting, Oct. 23, 2013-Nov. 1, 2013, Geneva, (Joint Collaborative Team on Video Coding of ISO/IEC JTC1/SC29/WG11 and ITU-TSG.16), No. JCTVC-00134, Oct. 22, 2013 (Oct. 22, 2013), XP030238512, 4 Pages, Retrieved from the Internet: URL: http://phenix.int-evry.fr/jct/doc_end_user/documents/15_Geneva/wg11/JCTVC-08134-v3.zip JCTVC-08134.doc. [retrieved on 2813-18-22] pp. 1-2.

Xu X "Description of Core Experiment 8 (CE8): Screen Content Coding Tools", Joint Video Experts Team (JVET) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11, Document: JVET-M1028-v4, 13th Meeting: Marrakech, MA, Jan. 9-18, 2019, pp. 1-18.

* cited by examiner

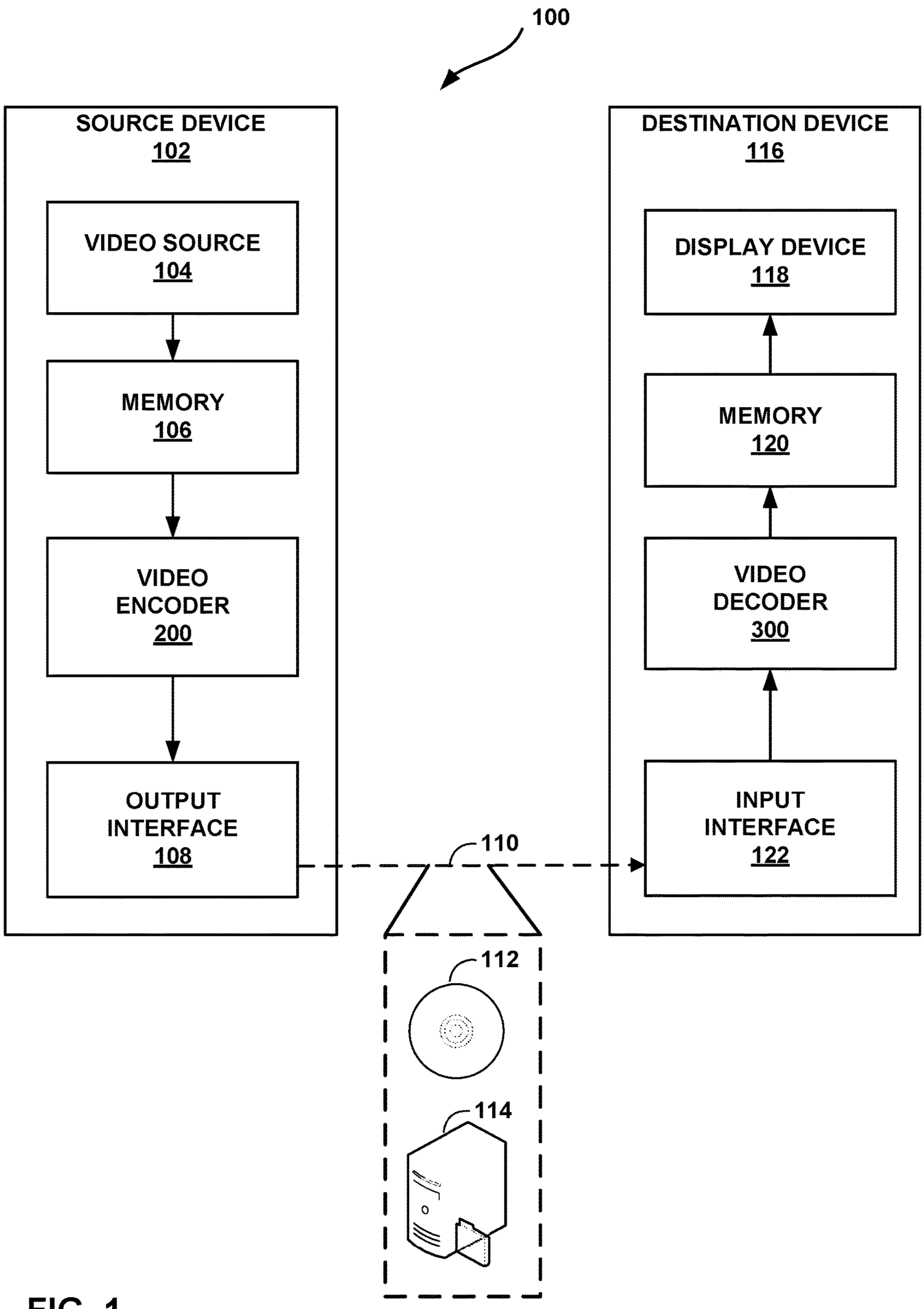

100
SOURCE DEVICE
102
VIDEO SOURCE
104
MEMORY
106
VIDEO
ENCODER
200
OUTPUT
INTERFACE
108
110
112
114
DESTINATION DEVICE
116
DISPLAY DEVICE
118
MEMORY
120
VIDEO
DECODER
300
INPUT
INTERFACE
122

FIG. 1

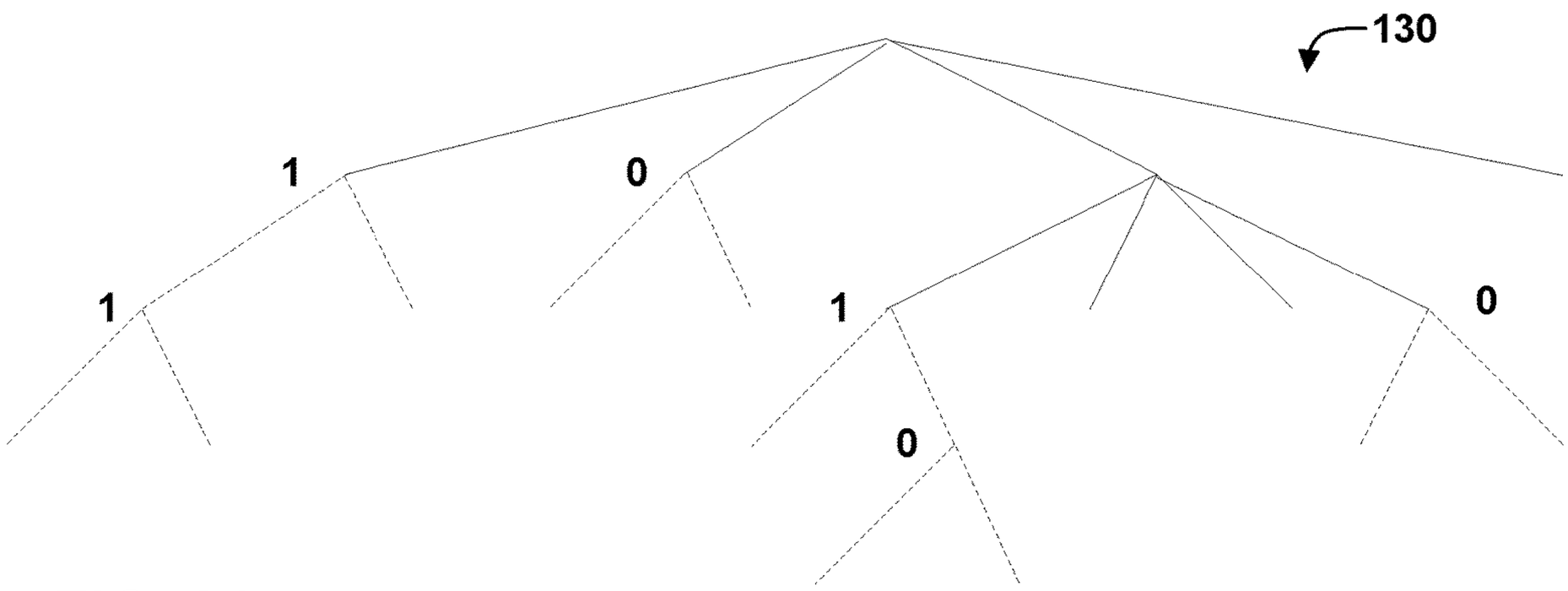


FIG. 2A

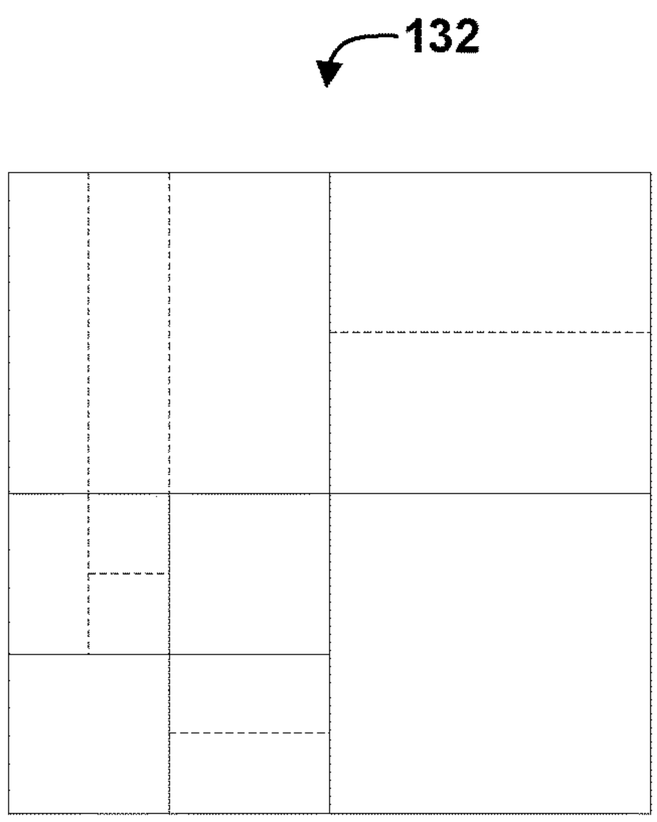


FIG. 2B

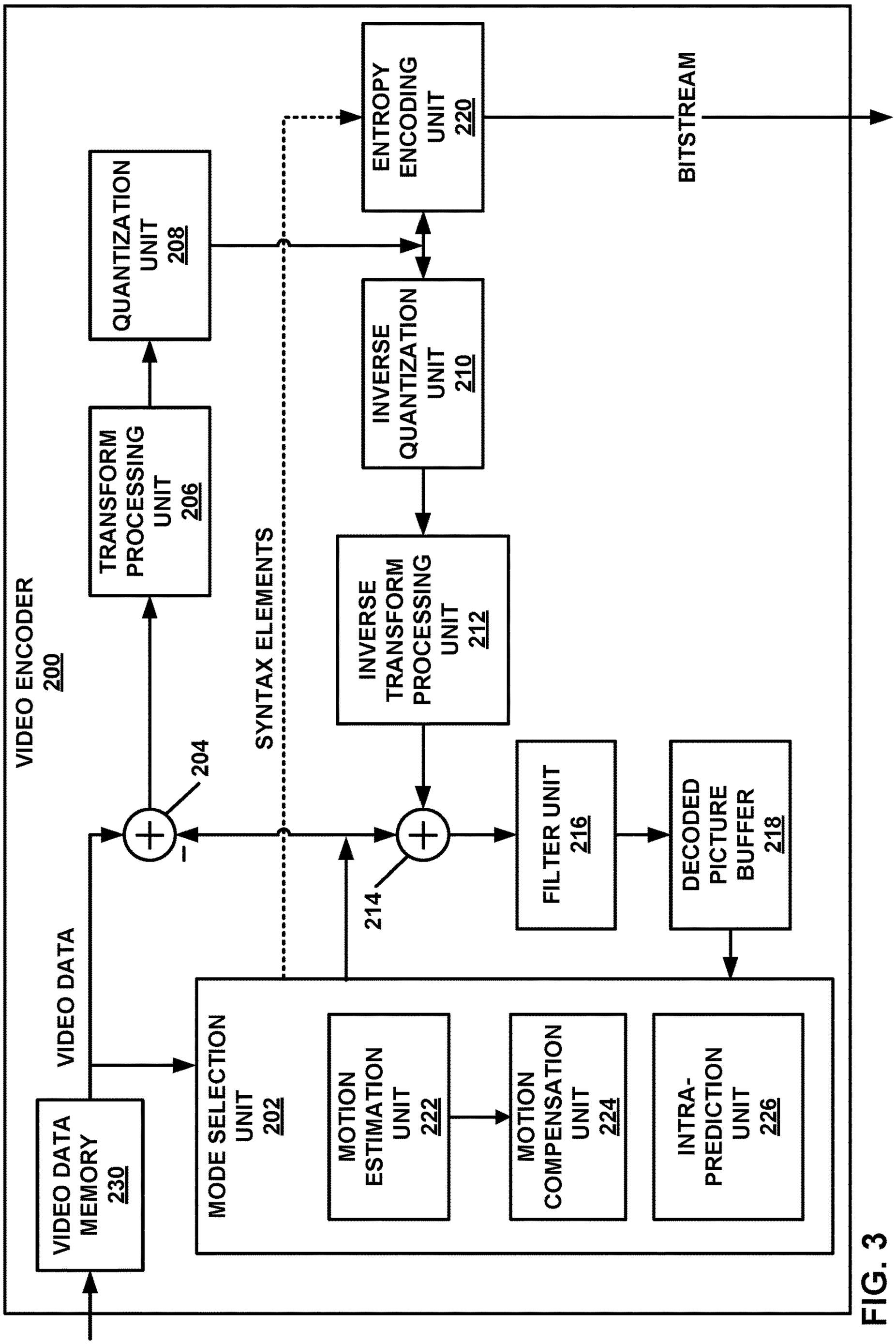

VIDEO ENCODER
200
VIDEO DATA
MEMORY
230
VIDEO DATA
MODE SELECTION
UNIT
202
MOTION
ESTIMATION
UNIT
222
MOTION
COMPENSATION
UNIT
224
INTRA-
PREDICTION
UNIT
226
TRANSFORM
PROCESSING
UNIT
206
QUANTIZATION
UNIT
208
INVERSE
QUANTIZATION
UNIT
210
INVERSE
TRANSFORM
PROCESSING
UNIT
212
ENTROPY
ENCODING
UNIT
220
BITSTREAM
204
214
SYNTAX ELEMENTS
FILTER UNIT
216
DECODED
PICTURE
BUFFER
218

FIG. 3

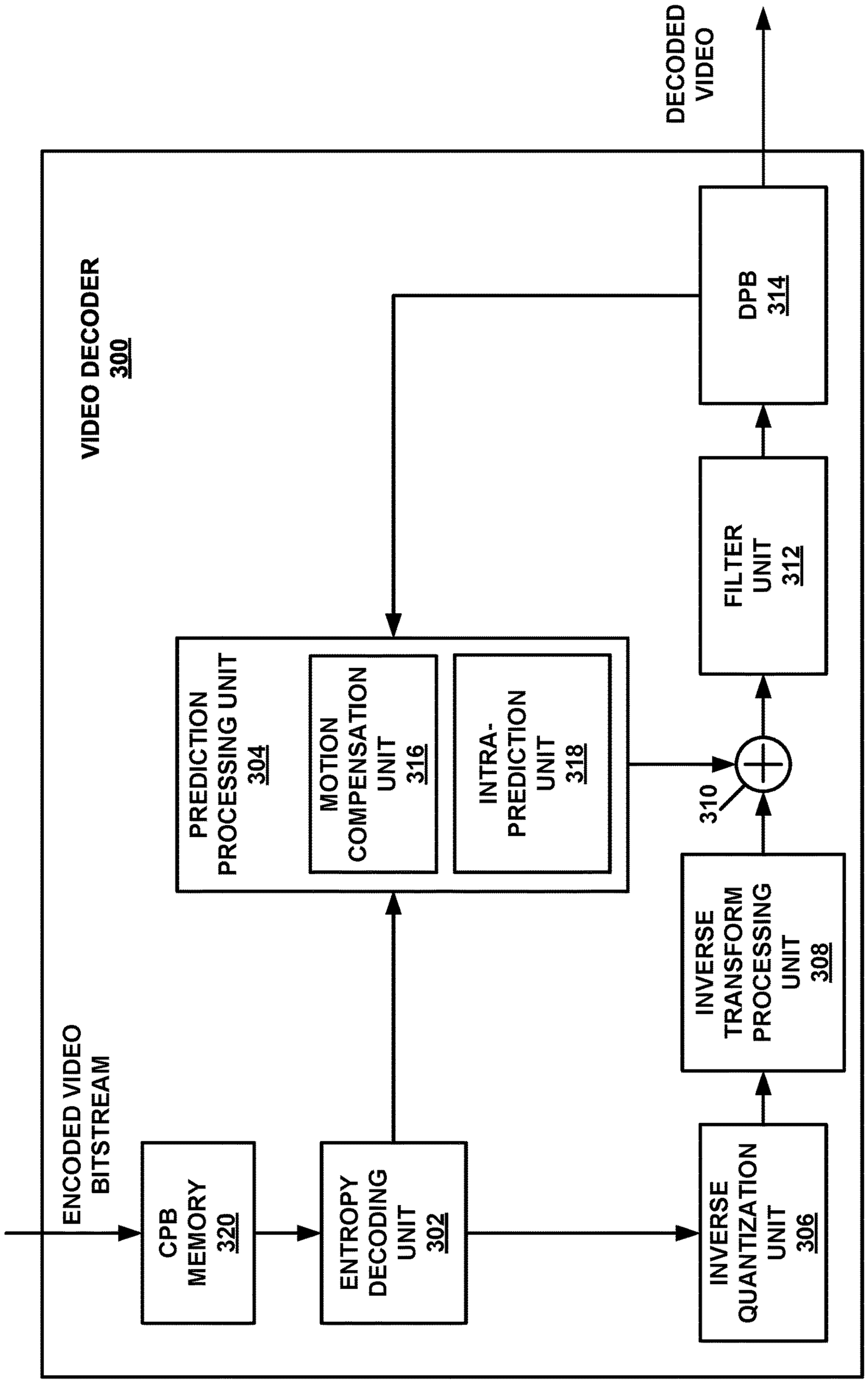

VIDEO DECODER
300
ENCODED VIDEO BITSTREAM
CPB MEMORY 320
ENTROPY DECODING UNIT 302
PREDICTION PROCESSING UNIT 304
MOTION COMPENSATION UNIT 316
INTRA-PREDICTION UNIT 318
INVERSE QUANTIZATION UNIT 306
INVERSE TRANSFORM PROCESSING UNIT 308
310
FILTER UNIT 312
DPB 314
DECODED VIDEO

FIG. 4

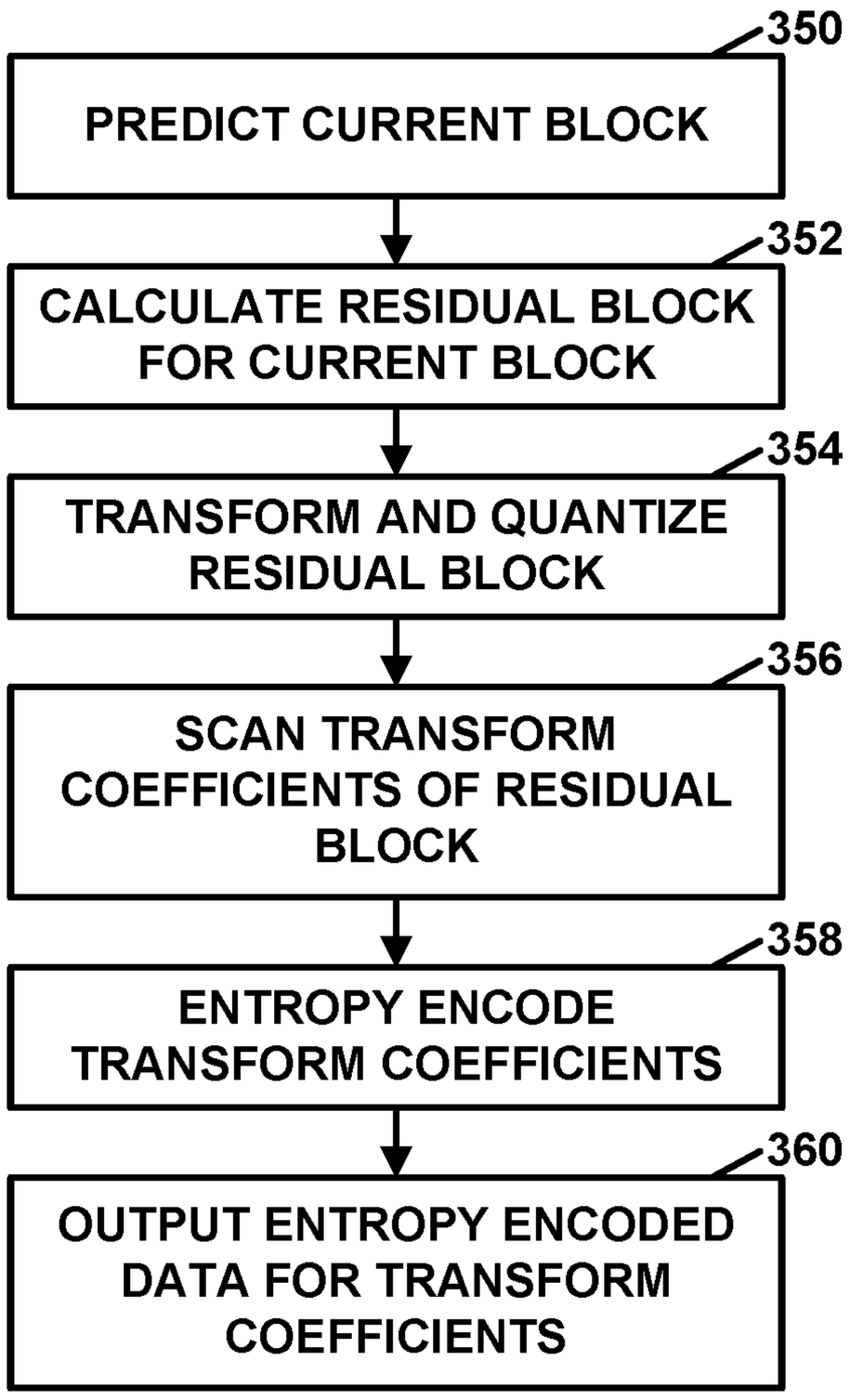

350
PREDICT CURRENT BLOCK
352
CALCULATE RESIDUAL BLOCK FOR CURRENT BLOCK
354
TRANSFORM AND QUANTIZE RESIDUAL BLOCK
356
SCAN TRANSFORM COEFFICIENTS OF RESIDUAL BLOCK
358
ENTROPY ENCODE TRANSFORM COEFFICIENTS
360
OUTPUT ENTROPY ENCODED DATA FOR TRANSFORM COEFFICIENTS

FIG. 5

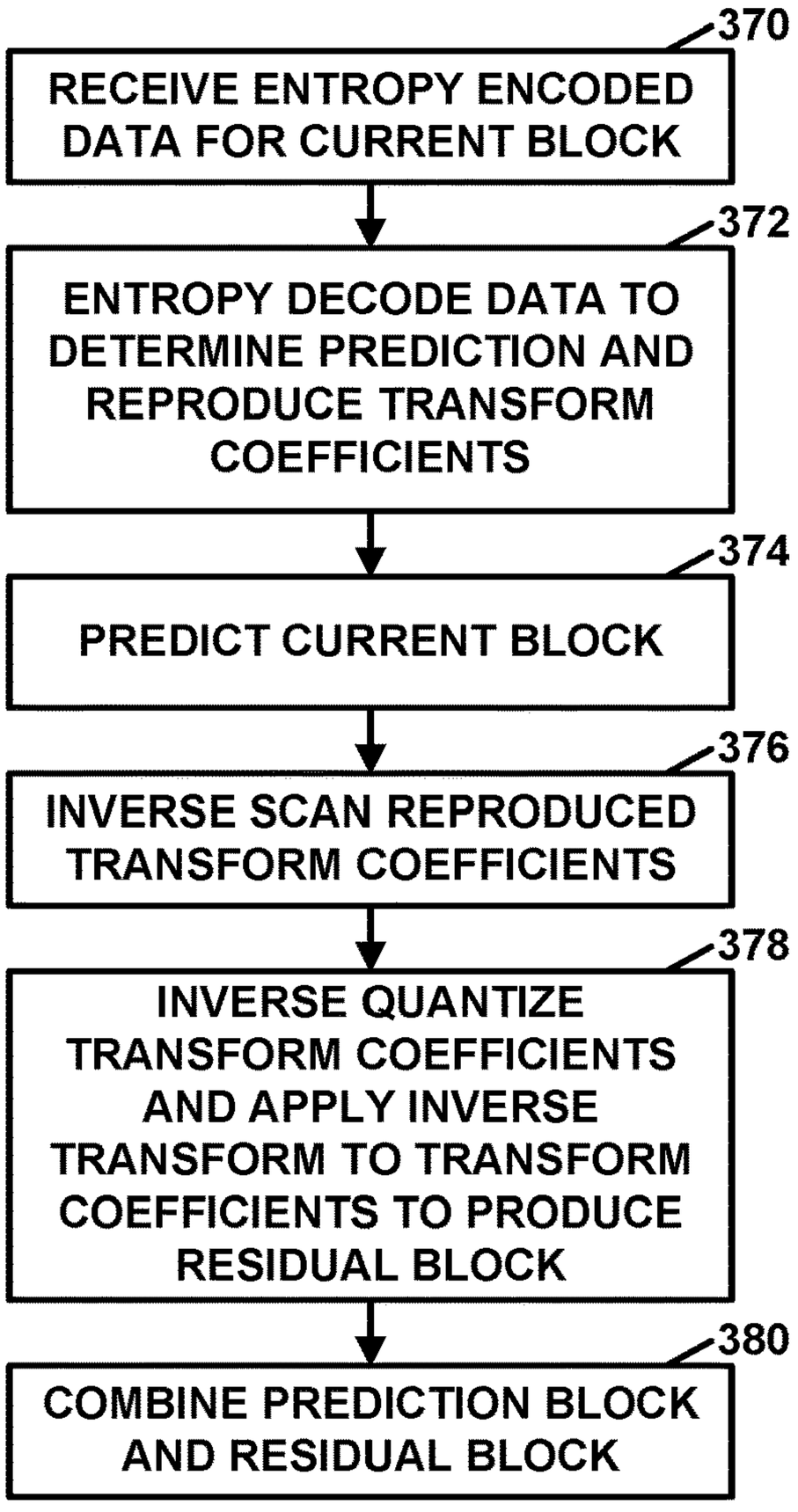

370
RECEIVE ENTROPY ENCODED DATA FOR CURRENT BLOCK
372
ENTROPY DECODE DATA TO DETERMINE PREDICTION AND REPRODUCE TRANSFORM COEFFICIENTS
374
PREDICT CURRENT BLOCK
376
INVERSE SCAN REPRODUCED TRANSFORM COEFFICIENTS
378
INVERSE QUANTIZE TRANSFORM COEFFICIENTS AND APPLY INVERSE TRANSFORM TO TRANSFORM COEFFICIENTS TO PRODUCE RESIDUAL BLOCK
380
COMBINE PREDICTION BLOCK AND RESIDUAL BLOCK

FIG. 6

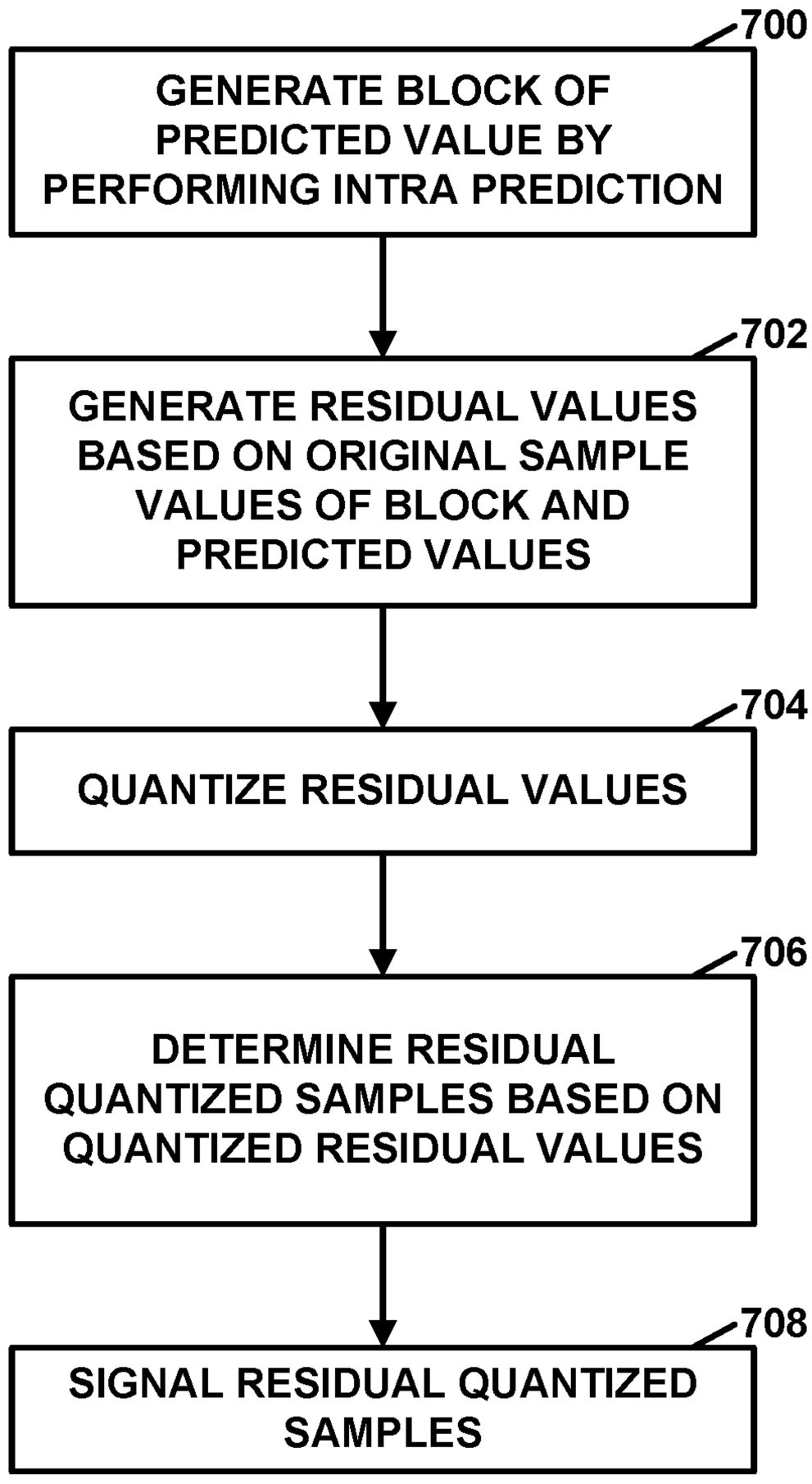

700
GENERATE BLOCK OF PREDICTED VALUE BY PERFORMING INTRA PREDICTION
702
GENERATE RESIDUAL VALUES BASED ON ORIGINAL SAMPLE VALUES OF BLOCK AND PREDICTED VALUES
704
QUANTIZE RESIDUAL VALUES
706
DETERMINE RESIDUAL QUANTIZED SAMPLES BASED ON QUANTIZED RESIDUAL VALUES
708
SIGNAL RESIDUAL QUANTIZED SAMPLES

FIG. 7

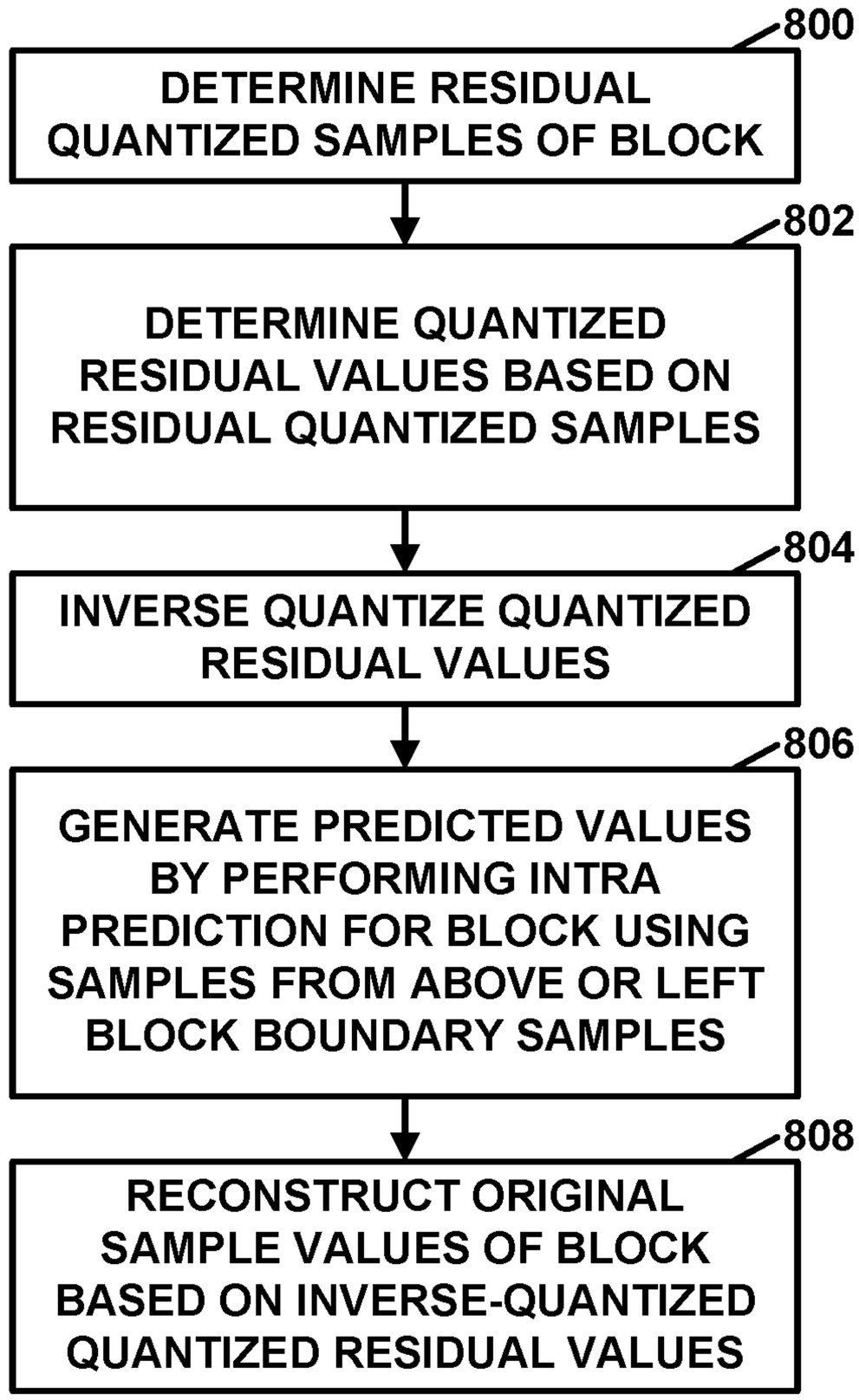

800
DETERMINE RESIDUAL QUANTIZED SAMPLES OF BLOCK
802
DETERMINE QUANTIZED RESIDUAL VALUES BASED ON RESIDUAL QUANTIZED SAMPLES
804
INVERSE QUANTIZE QUANTIZED RESIDUAL VALUES
806
GENERATE PREDICTED VALUES BY PERFORMING INTRA PREDICTION FOR BLOCK USING SAMPLES FROM ABOVE OR LEFT BLOCK BOUNDARY SAMPLES
808
RECONSTRUCT ORIGINAL SAMPLE VALUES OF BLOCK BASED ON INVERSE-QUANTIZED QUANTIZED RESIDUAL VALUES

FIG. 8

COEFFICIENT DOMAIN BLOCK DIFFERENTIAL PULSE-CODE MODULATION IN VIDEO CODING

This application claims the benefit of U.S. Provisional Patent Application 62/817,451, filed Mar. 12, 2019, the entire content of which is incorporated by reference.

TECHNICAL FIELD

This disclosure relates to video encoding and video decoding.

BACKGROUND

Digital video capabilities can be incorporated into a wide range of devices, including digital televisions, digital direct broadcast systems, wireless broadcast systems, personal digital assistants (PDAs), laptop or desktop computers, tablet computers, e-book readers, digital cameras, digital recording devices, digital media players, video gaming devices, video game consoles, cellular or satellite radio telephones, so-called "smart phones," video teleconferencing devices, video streaming devices, and the like. Digital video devices implement video coding techniques, such as those described in the standards defined by MPEG-2, MPEG-4, ITU-T H.263, ITU-T H.264/MPEG-4, Part 10, Advanced Video Coding (AVC), ITU-T H.265/High Efficiency Video Coding (HEVC), and extensions of such standards. The video devices may transmit, receive, encode, decode, and/or store digital video information more efficiently by implementing such video coding techniques.

Video coding techniques include spatial (intra-picture) prediction and/or temporal (inter-picture) prediction to reduce or remove redundancy inherent in video sequences. For block-based video coding, a video slice (e.g., a video picture or a portion of a video picture) may be partitioned into video blocks, which may also be referred to as coding tree units (CTUs), coding units (CUs) and/or coding nodes. Video blocks in an intra-coded (I) slice of a picture are encoded using spatial prediction with respect to reference samples in neighboring blocks in the same picture. Video blocks in an inter-coded (P or B) slice of a picture may use spatial prediction with respect to reference samples in neighboring blocks in the same picture or temporal prediction with respect to reference samples in other reference pictures. Pictures may be referred to as frames, and reference pictures may be referred to as reference frames.

SUMMARY

In general, this disclosure describes coefficient domain block differential pulse code modulation (BDPCM) methods and relates to techniques related to coefficient level prediction methods for transform skip coding of residual blocks in a video coding process. The corresponding entropy encoding process, which is the reverse process of entropy decoding, is implicitly specified and therefore is part of the techniques of this disclosure as well. The techniques of this disclosure may be applied to any of the existing video codecs, such as High Efficiency Video Coding (HEVC), or be applied as a coding tool to a video coding standard currently being developed, such as Versatile Video Coding (VVC), and/or to other future video coding standards.

In one example, this disclosure describes a method of decoding video data, the method comprising: determining, based on syntax elements in a bitstream that comprises an encoded representation of the video data, residual quantized samples of a block of the video data; determining quantized residual values based on the residual quantized samples; after determining the quantized residual values, inverse quantizing the quantized residual values; generating predicted values by performing intra prediction for the block using unfiltered samples from above or left block boundary samples; and reconstructing original sample values of the block based on the inverse-quantized quantized residual values and the prediction values.

In another example, this disclosure describes a method of encoding video data, the method comprising: generating a block of predicted values by performing intra prediction for a block of the video data using unfiltered samples from above or left block boundary samples; generating residual values based on original sample values of the block and the predicted values; quantizing the residual values; after quantizing the residual values, determining residual quantized samples based on the quantized residual values; and signaling the residual quantized samples.

In another example, this disclosure describes a device for decoding video data, the device comprising: a memory configured to store the video data; and one or more processors implemented in circuitry, the one or more processors configured to: determine, based on syntax elements in a bitstream that comprises an encoded representation of the video data, residual quantized samples of a block of the video data; determine quantized residual values based on the residual quantized samples; after determining the quantized residual values, inverse quantize the quantized residual values; generate predicted values by performing intra prediction for the block using unfiltered samples from above or left block boundary samples; and reconstruct original sample values of the block based on the inverse-quantized quantized residual values and the prediction values.

In another example, this disclosure describes a device for encoding video data, the device comprising: a memory configured to store the video data; and one or more processors implemented in circuitry, the one or more processors configured to: generate a block of predicted values by performing intra prediction for a block of the video data using unfiltered samples from above or left block boundary samples; generate residual values based on original sample values of the block and the predicted values; quantize the residual values; after quantizing the residual values, determine residual quantized samples based on the quantized residual values; and signal the residual quantized samples.

In another example, this disclosure describes a device for decoding video data, the device comprising: means for determining, based on syntax elements in a bitstream that comprises an encoded representation of the video data, residual quantized samples of a block of the video data; means for determining quantized residual values based on the residual quantized samples; means for inverse quantizing, after determining the quantized residual values, the quantized residual values; means for generating predicted values by performing intra prediction for the block using unfiltered samples from above or left block boundary samples; and means for reconstructing original sample values of the block based on the inverse-quantized quantized residual values and the prediction values.

In another example, this disclosure describes a device for encoding video data, the device comprising: means for generating a block of predicted values by performing intra prediction for a block of the video data using unfiltered samples from above or left block boundary samples; means for generating residual values based on original sample

values of the block and the predicted values; means for quantizing the residual values; means for determining, after quantizing the residual values, residual quantized samples based on the quantized residual values; and means for signaling the residual quantized samples.

In another example, this disclosure describes a computer-readable storage medium having stored thereon instructions that, when executed, cause one or more processors to: determine, based on syntax elements in a bitstream that comprises an encoded representation of the video data, residual quantized samples of a block of the video data; determine quantized residual values based on the residual quantized samples; after determining the quantized residual values, inverse quantize the quantized residual values; generate predicted values by performing intra prediction for the block using unfiltered samples from above or left block boundary samples; and reconstruct original sample values of the block based on the inverse-quantized quantized residual values and the prediction values.

In another example, this disclosure describes a computer-readable storage medium having stored thereon instructions that, when executed, cause one or more processors to: generate a block of predicted values by performing intra prediction for a block of the video data using unfiltered samples from above or left block boundary samples; generate residual values based on original sample values of the block and the predicted values; quantize the residual values; after quantizing the residual values, determine residual quantized samples based on the quantized residual values; and signal the residual quantized samples.

The details of one or more examples are set forth in the accompanying drawings and the description below. Other features, objects, and advantages will be apparent from the description, drawings, and claims.

BRIEF DESCRIPTION OF DRAWINGS

FIG. **1** is a block diagram illustrating an example video encoding and decoding system that may perform the techniques of this disclosure.

FIGS. **2**A and **2**B are conceptual diagrams illustrating an example quadtree binary tree (QTBT) structure, and a corresponding coding tree unit (CTU).

FIG. **3** is a block diagram illustrating an example video encoder that may perform the techniques of this disclosure.

FIG. **4** is a block diagram illustrating an example video decoder that may perform the techniques of this disclosure.

FIG. **5** is a flowchart illustrating a video encoding process.

FIG. **6** is a flowchart illustrating a video decoding process.

FIG. **7** is a flowchart illustrating an example video encoding process that includes coefficient domain block-differential pulse-code modulation (BDPCM), in accordance with one or more techniques of this disclosure.

FIG. **8** is a flowchart illustrating an example video decoding process that includes coefficient domain BDPCM, in accordance with one or more techniques of this disclosure.

DETAILED DESCRIPTION

In instances where a current block of video data is encoded using intra prediction, it may be advantageous to skip application of a transform that converts residual data of the current block from a sample domain to a frequency domain. Thus, the video encoder may quantize the residual data for the block directly. The video encoder may then include, in a bitstream, encoded syntax elements representing the quantized residual data.

Block-based delta pulse code modulation (BDPCM) may improve the encoding efficiency of the residual data of the current block. The video encoder may apply BDPCM in a vertical mode or a horizontal mode. When the video encoder applies BDPCM in the vertical mode, the video encoder generates predicted residual samples for a first row of the current block by subtracting original sample values in the first row of the current block from corresponding reconstructed samples in a bottom row of a neighboring block above the current block. The video encoder may then quantize and inverse quantize the predicted residual samples for the first row of the current block. The video encoder then reconstructs the first row of the current block based on the inverse quantized residual values for the first row of the current block and the reconstructed samples in the bottom row of the neighboring block above the current block. The video encoder may generate each subsequent row of predicted residual samples of the current block by subtracting original sample values of the row from the reconstructed samples of the row above in the current block. The video encoder may then perform the quantization, inverse quantization, and reconstruction process as before. The video encoder repeats this process for each row of the current block. For each row of the current block, the video encoder includes, in the bitstream, encoded syntax elements representing the quantized residual values for the row. In the vertical mode, the video encoder performs a similar process that works from left to right along columns of the current block.

A video decoder receives the encoded syntax elements representing the quantized residual values of the current block. When the current block is encoded using BDPCM and the vertical mode is used, the video decoder inverse quantizes the quantized residual values of a top row of the current block. The video decoder then reconstructs values of the top row of the current block by adding the inverse quantized residual values of the top row of the current block to corresponding reconstructed samples of a bottom row of one or more blocks that are above neighbors of the current block. For each respective subsequent rows of the current block, the video decoder inverse quantizes the quantized residual values for the respective row of the current block and then adds the inverse quantized residual values for the row to reconstructed samples of a row above the respective row of the current block, thereby reconstructing the samples of the respective row of the current block. When the current block is encoded using BDPCM and the horizontal mode is used, the video decoder performs a similar process that works from left to right along columns of the current block.

There may be one or more problems with the BDPCM process described above. For example, in the above-described BDPCM process (i.e., a pixel-domain BDPCM process), the video encoder subtracts samples in the current row or column from reconstructed samples in an adjacent row or column and the video decoder adds samples in the current row or column from reconstructed samples in an adjacent row or column. It is appreciated in this disclosure that the use of reconstructed samples may slow down and complicate the encoding and decoding processes because the video encoder and video decoder may need to wait for inverse quantization and reconstruction to occur before being able to determine a predictor for a current row or column of the current block. Hence, in accordance with a technique of this disclosure, the video encoder and the video decoder may determine the predicted residual values using quantized residual values instead of reconstructed samples.

For instance, in one example in accordance with the techniques of this disclosure, a video encoder may generate a block of predicted values by performing intra prediction for a block of the video data using samples (e.g., unfiltered samples) from above or left block boundary samples. In this example, the video encoder may generate residual values based on original sample values of the block and the predicted values. Furthermore, the video encoder may quantize the residual values. After quantizing the residual values, the video encoder may determine residual quantized samples based on the quantized residual values. The video encoder may signal the residual quantized samples.

In another example in accordance with the techniques of this disclosure, a video decoder may determine, based on syntax elements in a bitstream that comprises an encoded representation of the video data, residual quantized samples of a block of the video data. Additionally, the video decoder may determine quantized residual values based on the residual quantized samples. After determining the quantized residual values, the video decoder may inverse quantize the quantized residual values. The video decoder may also generate predicted values by performing intra prediction for the block using unfiltered samples from above or left block boundary samples. The video decoder may reconstruct original sample values of the block based on the inverse-quantized quantized residual values and the prediction values.

FIG. **1** is a block diagram illustrating an example video encoding and decoding system **100** that may perform the techniques of this disclosure. The techniques of this disclosure are generally directed to coding (encoding and/or decoding) video data. In general, video data includes any data for processing a video. Thus, video data may include raw, unencoded video, encoded video, decoded (e.g., reconstructed) video, and video metadata, such as signaling data.

As shown in FIG. **1**, system **100** includes a source device **102** that provides encoded video data to be decoded and displayed by a destination device **116**, in this example. In particular, source device **102** provides the video data to destination device **116** via a computer-readable medium **110**. Source device **102** and destination device **116** may comprise any of a wide range of devices, including desktop computers, notebook (i.e., laptop) computers, tablet computers, set-top boxes, telephone handsets such smartphones, televisions, cameras, display devices, computers, mobile devices, broadcast receiver devices, digital media players, video gaming consoles, video streaming device, or the like. In some cases, source device **102** and destination device **116** may be equipped for wireless communication, and thus may be referred to as wireless communication devices.

In the example of FIG. **1**, source device **102** includes video source **104**, memory **106**, video encoder **200**, and output interface **108**. Destination device **116** includes input interface **122**, video decoder **300**, memory **120**, and display device **118**. In accordance with this disclosure, video encoder **200** of source device **102** and video decoder **300** of destination device **116** may be configured to apply the techniques for coefficient level prediction. Thus, source device **102** represents an example of a video encoding device, while destination device **116** represents an example of a video decoding device. In other examples, a source device and a destination device may include other components or arrangements. For example, source device **102** may receive video data from an external video source, such as an external camera. Likewise, destination device **116** may interface with an external display device, rather than including an integrated display device.

System **100** as shown in FIG. **1** is merely one example. In general, any digital video encoding and/or decoding device may perform techniques for coefficient level prediction. Source device **102** and destination device **116** are merely examples of such coding devices in which source device **102** generates coded video data for transmission to destination device **116**. This disclosure refers to a "coding" device as a device that performs coding (encoding and/or decoding) of data. Thus, video encoder **200** and video decoder **300** represent examples of coding devices, in particular, a video encoder and a video decoder, respectively. In some examples, source device **102** and destination device **116** may operate in a substantially symmetrical manner such that each of source device **102** and destination device **116** includes video encoding and decoding components. Hence, system **100** may support one-way or two-way video transmission between source device **102** and destination device **116**, e.g., for video streaming, video playback, video broadcasting, or video telephony.

In general, video source **104** represents a source of video data (i.e., raw, uncoded video data) and provides a sequential series of pictures (also referred to as "frames") of the video data to video encoder **200**, which encodes data for the pictures. Video source **104** of source device **102** may include a video capture device, such as a video camera, a video archive containing previously captured raw video, and/or a video feed interface to receive video from a video content provider. As a further alternative, video source **104** may generate computer graphics-based data as the source video, or a combination of live video, archived video, and computer-generated video. In each case, video encoder **200** encodes the captured, pre-captured, or computer-generated video data. Video encoder **200** may rearrange the pictures from the received order (sometimes referred to as "display order") into a coding order for coding. Video encoder **200** may generate a bitstream including encoded video data. Source device **102** may then output the encoded video data via output interface **108** onto computer-readable medium **110** for reception and/or retrieval by, e.g., input interface **122** of destination device **116**.

Memory **106** of source device **102** and memory **120** of destination device **116** represent general purpose memories. In some example, memories **106**, **120** may store raw video data, e.g., raw video from video source **104** and raw, decoded video data from video decoder **300**. Additionally or alternatively, memories **106**, **120** may store software instructions executable by, e.g., video encoder **200** and video decoder **300**, respectively. Although memory **106** and memory **120** are shown separately from video encoder **200** and video decoder **300** in this example, it should be understood that video encoder **200** and video decoder **300** may also include internal memories for functionally similar or equivalent purposes. Furthermore, memories **106**, **120** may store encoded video data, e.g., output from video encoder **200** and input to video decoder **300**. In some examples, portions of memories **106**, **120** may be allocated as one or more video buffers, e.g., to store raw, decoded, and/or encoded video data.

Computer-readable medium **110** may represent any type of medium or device capable of transporting the encoded video data from source device **102** to destination device **116**. In one example, computer-readable medium **110** represents a communication medium to enable source device **102** to transmit encoded video data directly to destination device **116** in real-time, e.g., via a radio frequency network or computer-based network. Output interface **108** may modulate a transmission signal including the encoded video data,

and input interface **122** may demodulate the received transmission signal, according to a communication standard, such as a wireless communication protocol. The communication medium may comprise any wireless or wired communication medium, such as a radio frequency (RF) spectrum or one or more physical transmission lines. The communication medium may form part of a packet-based network, such as a local area network, a wide-area network, or a global network such as the Internet. The communication medium may include routers, switches, base stations, or any other equipment that may be useful to facilitate communication from source device **102** to destination device **116**.

In some examples, computer-readable medium **110** may include storage device **112**. Source device **102** may output encoded data from output interface **108** to storage device **112**. Similarly, destination device **116** may access encoded data from storage device **112** via input interface **122**. Storage device **112** may include any of a variety of distributed or locally accessed data storage media such as a hard drive, Blu-ray discs, DVDs, CD-ROMs, flash memory, volatile or non-volatile memory, or any other suitable digital storage media for storing encoded video data.

In some examples, computer-readable medium **110** may include file server **114** or another intermediate storage device that may store the encoded video data generated by source device **102**. Source device **102** may output encoded video data to file server **114** or another intermediate storage device that may store the encoded video generated by source device **102**. Destination device **116** may access stored video data from file server **114** via streaming or download. File server **114** may be any type of server device capable of storing encoded video data and transmitting that encoded video data to the destination device **116**. File server **114** may represent a web server (e.g., for a website), a File Transfer Protocol (FTP) server, a content delivery network device, or a network attached storage (NAS) device. Destination device **116** may access encoded video data from file server **114** through any standard data connection, including an Internet connection. This may include a wireless channel (e.g., a Wi-Fi connection), a wired connection (e.g., digital subscriber line (DSL), cable modem, etc.), or a combination of both that is suitable for accessing encoded video data stored on file server **114**. File server **114** and input interface **122** may be configured to operate according to a streaming transmission protocol, a download transmission protocol, or a combination thereof.

Output interface **108** and input interface **122** may represent wireless transmitters/receiver, modems, wired networking components (e.g., Ethernet cards), wireless communication components that operate according to any of a variety of IEEE 802.11 standards, or other physical components. In examples where output interface **108** and input interface **122** comprise wireless components, output interface **108** and input interface **122** may be configured to transfer data, such as encoded video data, according to a cellular communication standard, such as 4G, 4G-LTE (Long-Term Evolution), LTE Advanced, 5G, or the like. In some examples where output interface **108** comprises a wireless transmitter, output interface **108** and input interface **122** may be configured to transfer data, such as encoded video data, according to other wireless standards, such as an IEEE 802.11 specification, an IEEE 802.15 specification (e.g., ZigBee™), a Bluetooth™ standard, or the like. In some examples, source device **102** and/or destination device **116** may include respective system-on-a-chip (SoC) devices. For example, source device **102** may include an SoC device to perform the functionality attributed to video encoder **200** and/or output interface **108**, and destination device **116** may include an SoC device to perform the functionality attributed to video decoder **300** and/or input interface **122**.

The techniques of this disclosure may be applied to video coding in support of any of a variety of multimedia applications, such as over-the-air television broadcasts, cable television transmissions, satellite television transmissions, Internet streaming video transmissions, such as dynamic adaptive streaming over HTTP (DASH), digital video that is encoded onto a data storage medium, decoding of digital video stored on a data storage medium, or other applications.

Input interface **122** of destination device **116** receives an encoded video bitstream from computer-readable medium **110** (e.g., a communication medium, storage device **112**, file server **114**, or the like). The encoded video bitstream may include signaling information defined by video encoder **200**, which is also used by video decoder **300**, such as syntax elements having values that describe characteristics and/or processing of video blocks or other coded units (e.g., slices, pictures, groups of pictures, sequences, or the like). Display device **118** displays decoded pictures of the decoded video data to a user. Display device **118** may represent any of a variety of display devices such as a cathode ray tube (CRT), a liquid crystal display (LCD), a plasma display, an organic light emitting diode (OLED) display, or another type of display device.

Although not shown in FIG. **1**, in some examples, video encoder **200** and video decoder **300** may each be integrated with an audio encoder and/or audio decoder, and may include appropriate MUX-DEMUX units, or other hardware and/or software, to handle multiplexed streams including both audio and video in a common data stream. If applicable, MUX-DEMUX units may conform to the ITU H.223 multiplexer protocol, or other protocols such as the user datagram protocol (UDP).

Video encoder **200** and video decoder **300** each may be implemented as any of a variety of suitable encoder and/or decoder circuitry, such as one or more microprocessors, digital signal processors (DSPs), application specific integrated circuits (ASICs), field programmable gate arrays (FPGAs), discrete logic, software, hardware, firmware or any combinations thereof. When the techniques are implemented partially in software, a device may store instructions for the software in a suitable, non-transitory computer-readable medium and execute the instructions in hardware using one or more processors to perform the techniques of this disclosure. Each of video encoder **200** and video decoder **300** may be included in one or more encoders or decoders, either of which may be integrated as part of a combined encoder/decoder (CODEC) in a respective device. A device including video encoder **200** and/or video decoder **300** may comprise an integrated circuit, a microprocessor, and/or a wireless communication device, such as a cellular telephone.

Video encoder **200** and video decoder **300** may operate according to a video coding standard, such as ITU-T H.265, also referred to as High Efficiency Video Coding (HEVC) or extensions thereto, such as the multi-view and/or scalable video coding extensions. Alternatively, video encoder **200** and video decoder **300** may operate according to other proprietary or industry standards, such as the Joint Exploration Test Model (JEM) or ITU-T H.266, also referred to as Versatile Video Coding (VVC). A recent draft of the VVC standard is described in Bross, et al. "Versatile Video Coding (Draft 4)," Joint Video Experts Team (JVET) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11, 13*th* Meeting: Marrakech, MA, 9-18 Jan. 2019, JVET-M1001-v5 (herein-

after "VVC Draft 4"). The techniques of this disclosure, however, are not limited to any particular coding standard.

In general, video encoder **200** and video decoder **300** may perform block-based coding of pictures. The term "block" generally refers to a structure including data to be processed (e.g., encoded, decoded, or otherwise used in the encoding and/or decoding process). For example, a block may include a two-dimensional matrix of samples of luminance and/or chrominance data. In general, video encoder **200** and video decoder **300** may code video data represented in a YUV (e.g., Y, Cb, Cr) format. That is, rather than coding red, green, and blue (RGB) data for samples of a picture, video encoder **200** and video decoder **300** may code luminance and chrominance components, where the chrominance components may include both red hue and blue hue chrominance components. In some examples, video encoder **200** converts received RGB formatted data to a YUV representation prior to encoding, and video decoder **300** converts the YUV representation to the RGB format. Alternatively, pre- and post-processing units (not shown) may perform these conversions.

This disclosure may generally refer to coding (e.g., encoding and decoding) of pictures to include the process of encoding or decoding data of the picture. Similarly, this disclosure may refer to coding of blocks of a picture to include the process of encoding or decoding data for the blocks, e.g., prediction and/or residual coding. An encoded video bitstream generally includes a series of values for syntax elements representative of coding decisions (e.g., coding modes) and partitioning of pictures into blocks. Thus, references to coding a picture or a block should generally be understood as coding values for syntax elements forming the picture or block.

HEVC defines various blocks, including coding units (CUs), prediction units (PUs), and transform units (TUs). According to HEVC, a video coder (such as video encoder **200**) partitions a coding tree unit (CTU) into CUs according to a quadtree structure. That is, the video coder partitions CTUs and CUs into four equal, non-overlapping squares, and each node of the quadtree has either zero or four child nodes. Nodes without child nodes may be referred to as "leaf nodes," and CUs of such leaf nodes may include one or more PUs and/or one or more TUs. The video coder may further partition PUs and TUs. For example, in HEVC, a residual quadtree (RQT) represents partitioning of TUs. In HEVC, PUs represent inter-prediction data, while TUs represent residual data. CUs that are intra-predicted include intra-prediction information, such as an intra-mode indication.

As another example, video encoder **200** and video decoder **300** may be configured to operate according to JEM or VVC. According to JEM or VVC, a video coder (such as video encoder **200**) partitions a picture into a plurality of coding tree units (CTUs). Video encoder **200** may partition a CTU according to a tree structure, such as a quadtree-binary tree (QTBT) structure or Multi-Type Tree (MTT) structure. The QTBT structure removes the concepts of multiple partition types, such as the separation between CUs, PUs, and TUs of HEVC. A QTBT structure includes two levels: a first level partitioned according to quadtree partitioning, and a second level partitioned according to binary tree partitioning. A root node of the QTBT structure corresponds to a CTU. Leaf nodes of the binary trees correspond to coding units (CUs).

In an MTT partitioning structure, blocks may be partitioned using a quadtree (QT) partition, a binary tree (BT) partition, and one or more types of triple tree (TT) partitions. A triple tree partition is a partition where a block is split into three sub-blocks. In some examples, a triple tree partition divides a block into three sub-blocks without dividing the original block through the center. The partitioning types in MTT (e.g., QT, BT, and TT), may be symmetrical or asymmetrical.

In some examples, video encoder **200** and video decoder **300** may use a single QTBT or MTT structure to represent each of the luminance and chrominance components, while in other examples, video encoder **200** and video decoder **300** may use two or more QTBT or MTT structures, such as one QTBT/MTT structure for the luminance component and another QTBT/MTT structure for both chrominance components (or two QTBT/MTT structures for respective chrominance components).

Video encoder **200** and video decoder **300** may be configured to use quadtree partitioning per HEVC, QTBT partitioning, MTT partitioning, or other partitioning structures. For purposes of explanation, the description of the techniques of this disclosure is presented with respect to QTBT partitioning. However, it should be understood that the techniques of this disclosure may also be applied to video coders configured to use quadtree partitioning, or other types of partitioning as well.

This disclosure may use "N×N" and "N by N" interchangeably to refer to the sample dimensions of a block (such as a CU or other video block) in terms of vertical and horizontal dimensions, e.g., 16×16 samples or 16 by 16 samples. In general, a 16×16 CU will have 16 samples in a vertical direction (y=16) and 16 samples in a horizontal direction (x=16). Likewise, an N×N CU generally has N samples in a vertical direction and N samples in a horizontal direction, where N represents a nonnegative integer value. The samples in a CU may be arranged in rows and columns. Moreover, CUs need not necessarily have the same number of samples in the horizontal direction as in the vertical direction. For example, CUs may comprise N×M samples, where M is not necessarily equal to N.

Video encoder **200** encodes video data for CUs representing prediction and/or residual information, and other information. The prediction information indicates how the CU is to be predicted in order to form a prediction block for the CU. The residual information generally represents sample-by-sample differences between samples of the CU prior to encoding and the prediction block.

To predict a CU, video encoder **200** may generally form a prediction block for the CU through inter-prediction or intra-prediction. Inter-prediction generally refers to predicting the CU from data of a previously coded picture, whereas intra-prediction generally refers to predicting the CU from previously coded data of the same picture. To perform inter-prediction, video encoder **200** may generate the prediction block using one or more motion vectors. Video encoder **200** may generally perform a motion search to identify a reference block that closely matches the CU, e.g., in terms of differences between the CU and the reference block. Video encoder **200** may calculate a difference metric using a sum of absolute difference (SAD), sum of squared differences (SSD), mean absolute difference (MAD), mean squared differences (MSD), or other such difference calculations to determine whether a reference block closely matches the current CU. In some examples, video encoder **200** may predict the current CU using uni-directional prediction or bi-directional prediction.

Some examples of JEM and VVC also provide an affine motion compensation mode, which may be considered an inter-prediction mode. In affine motion compensation mode, video encoder **200** may determine two or more motion

vectors that represent non-translational motion, such as zoom in or out, rotation, perspective motion, or other irregular motion types.

To perform intra-prediction, video encoder **200** may select an intra prediction mode to generate the prediction block. Some examples of JEM and VVC provide sixty-seven intra prediction modes, including various directional modes, as well as planar mode and DC mode. In general, video encoder **200** selects an intra prediction mode that describes neighboring samples to a current block (e.g., a block of a CU) from which to predict samples of the current block. Accordingly, such neighboring samples may be referred to as a predictor. Such samples may generally be above, above and to the left, or to the left of the current block in the same picture as the current block, assuming video encoder **200** codes CTUs and CUs in raster scan order (left to right, top to bottom).

Directional intra prediction modes correspond to different directions, including a vertical direction and a horizontal direction. To generate a prediction block for a block using a vertical intra prediction mode, a video coder (e.g., video encoder **200** or video decoder **300**) may, for each sample of the block, determine a predicted value of the sample as the sample in the predictor that is directly above the sample. To generate a prediction block for a block using a horizontal intra prediction, a video coder (e.g., video encoder **200** or video decoder **300**) may, for each sample of the block, determine a predicted value of the sample as the sample in the predictor that is directly left of the sample.

In some examples, a video coder may apply one or more filters to a predictor before using the predictor for intra prediction of a block. For instance, the video coder may apply a smoothing filter to the predictor. Application of such filters may improve coding efficiency in some situations. However, in other situations, it may be advantageous not to apply such filters. Accordingly, in situations where no filter is applied to the predictor used in intra prediction of a block, the samples of the predictor may be referred to in this disclosure as unfiltered samples.

Video encoder **200** encodes data representing the prediction mode for a current block. For example, for inter-prediction modes, video encoder **200** may encode data representing which of the various available inter-prediction modes is used, as well as motion information for the corresponding mode. For uni-directional or bi-directional inter-prediction, for example, video encoder **200** may encode motion vectors using advanced motion vector prediction (AMVP) or merge mode. Video encoder **200** may use similar modes to encode motion vectors for affine motion compensation mode.

Following prediction, such as intra-prediction or inter-prediction of a block, video encoder **200** may calculate residual data for the block. The residual data, such as a residual block, represents sample by sample differences between the block and a prediction block for the block, formed using the corresponding prediction mode. Video encoder **200** may apply one or more transforms to the residual block, to produce transformed data in a transform domain instead of the sample domain. For example, video encoder **200** may apply a discrete cosine transform (DCT), an integer transform, a wavelet transform, or a conceptually similar transform to residual video data. Additionally, video encoder **200** may apply a secondary transform following the first transform, such as a mode-dependent non-separable secondary transform (MDNSST), a signal dependent transform, a Karhunen-Loeve transform (KLT), or the like.

Video encoder **200** produces transform coefficients following application of the one or more transforms. In some examples, video encoder **200** may skip application of the transform. In such cases, residual data may be processed in a similar manner as the transform coefficients generated by application of the transform. For ease of explanation, steps occurring at video encoder **200** after the point where the transform is applied may be referred to as the transform domain, regardless of whether video encoder **200** actually applied the transform. For instance, video encoder **200** does not apply the transform when video encoder **200** applies BDPCM. Similarly, steps occurring at video decoder **300** before the point where an inverse of the transform is applied may be referred to as the transform domain, regardless of whether the transform was actually applied. For instance, video decoder **300** does not apply the inverse transform when video decoder **300** applies BDPCM.

As noted above, following any transforms to produce transform coefficients, video encoder **200** may perform quantization of the transform coefficients. Quantization generally refers to a process in which transform coefficients are quantized to possibly reduce the amount of data used to represent the transform coefficients, providing further compression. By performing the quantization process, video encoder **200** may reduce the bit depth associated with some or all of the transform coefficients. For example, video encoder **200** may round an n-bit value down to an m-bit value during quantization, where n is greater than m. In some examples, to perform quantization, video encoder **200** may perform a bitwise right-shift of the value to be quantized.

Following quantization, video encoder **200** may scan the transform coefficients, producing a one-dimensional vector from the two-dimensional matrix including the quantized transform coefficients. The scan may be designed to place higher energy (and therefore lower frequency) transform coefficients at the front of the vector and to place lower energy (and therefore higher frequency) transform coefficients at the back of the vector. In some examples, video encoder **200** may utilize a predefined scan order to scan the quantized transform coefficients to produce a serialized vector, and then entropy encode the quantized transform coefficients of the vector. In other examples, video encoder **200** may perform an adaptive scan. After scanning the quantized transform coefficients to form the one-dimensional vector, video encoder **200** may entropy encode the one-dimensional vector, e.g., according to context-adaptive binary arithmetic coding (CABAC). Video encoder **200** may also entropy encode values for syntax elements describing metadata associated with the encoded video data for use by video decoder **300** in decoding the video data.

To perform CABAC, video encoder **200** may assign a context within a context model to a symbol to be transmitted. The context may relate to, for example, whether neighboring values of the symbol are zero-valued or not. The probability determination may be based on a context assigned to the symbol.

Video encoder **200** may further generate syntax data, such as block-based syntax data, picture-based syntax data, and sequence-based syntax data, to video decoder **300**, e.g., in a picture header, a block header, a slice header, or other syntax data, such as a sequence parameter set (SPS), picture parameter set (PPS), or video parameter set (VPS). Video decoder **300** may likewise decode such syntax data to determine how to decode corresponding video data.

In this manner, video encoder **200** may generate a bitstream including encoded video data, e.g., syntax elements

describing partitioning of a picture into blocks (e.g., CUs) and prediction and/or residual information for the blocks. Ultimately, video decoder **300** may receive the bitstream and decode the encoded video data.

In general, video decoder **300** performs a reciprocal process to that performed by video encoder **200** to decode the encoded video data of the bitstream. For example, video decoder **300** may decode values for syntax elements of the bitstream using CABAC in a manner substantially similar to, albeit reciprocal to, the CABAC encoding process of video encoder **200**. The syntax elements may define partitioning information for partitioning a picture into CTUs, and partitioning of each CTU according to a corresponding partition structure, such as a QTBT structure, to define CUs of the CTU. The syntax elements may further define prediction and residual information for blocks (e.g., CUs) of video data.

The residual information may be represented by, for example, quantized transform coefficients. Video decoder **300** may inverse quantize and inverse transform the quantized transform coefficients of a block to reproduce a residual block for the block. Video decoder **300** uses a signaled prediction mode (intra- or inter-prediction) and related prediction information (e.g., motion information for inter-prediction) to form a prediction block for the block. Video decoder **300** may then combine the prediction block and the residual block (on a sample-by-sample basis) to reproduce the original block. Video decoder **300** may perform additional processing, such as performing a deblocking process to reduce visual artifacts along boundaries of the block.

Abdoli et al., "CE8: BDPCM with horizontal/vertical predictor and independently decodable areas (test 8.3.1b)," Joint Video Experts Team (JVET) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11, 13th Meeting, Marrakech, MA, January 2019, document no. JVET-M0057 (hereinafter, "JVET-M0057") described a Block-Differential Pulse-Code Modulation (DPCM) mode that utilizes horizontal or vertical prediction of intra samples that is combined with DPCM coding and transform skip coding. On the encoder side, for vertical prediction, a top neighboring block's bottom row pixels are used to perform vertical intra prediction of the first horizontal line of the block with unfiltered predictor samples. Predicted residuals are quantized, inverse quantized and added to the predictor to form the predictor for the prediction of the next line. Video encoder **200** continues this until the end of the block. For horizontal prediction, a similar scheme applies except the initial predictor is from the left neighboring block's last column, the first column of the block to be coded is predicted, and the residuals are quantized, inverse quantized and added to the predictor to form the predictor of the next column. Additional, details are described in JVET-M0057. The direction (horizontal/vertical) of BDPCM prediction may be signaled at a CU level.

Instead of doing the prediction in a pixel domain, as described in WET-M0057, using reconstructed samples from above or left, the present disclosure describes quantized level domain prediction in the same direction. This can be described as follows.

Consider a block of size M (rows)×N (cols). Let $r_{i,j}$, $0 \le i \le M-1, 0 \le j \le N-1$ be the prediction residual after performing intra prediction horizontally (copying left neighbor pixel value across the predicted block line by line) or vertically (copying top neighbor line to each line in the predicted block) using unfiltered samples from above or left block boundary samples. Assume that the transform is skipped and let $Q(r_{i,j})$, $0 \le i \le M-1, 0 \le j \le N-1$ denote the quantized version of the residual $r_{i,j}$, where the residual is the difference between the original block and the predicted block values. Video encoder **200** may apply the BDPCM to the quantized residual samples as follows: Video encoder **200** may obtain the modified M×N array $\tilde{R}$ with elements $\tilde{r}_{i,j}$ according to equation (1) as follows when vertical BDPCM is signalled:

$$\tilde{r}_{i,j} = \begin{cases} Q(r_{i,j}), & i = 0, 0 \le j \le (N-1) \\ Q(r_{i,j}) - Q(r_{(i-1),j}), & 1 \le i \le (M-1), 0 \le j \le (N-1) \end{cases} \quad (1)$$

For horizontal prediction, similar rules apply, and video encoder **200** may obtain the residual quantized samples by equation (2) as follows:

$$\tilde{r}_{i,j} = \begin{cases} Q(r_{i,j}), & 0 \le i \le (M-1), j = 0 \\ Q(r_{i,j}) - Q(r_{i,(j-1)}), & 0 \le i \le (M-1), 1 \le j \le (N-1) \end{cases} \quad (2)$$

Video encoder **200** may send the residual quantized samples $\tilde{r}_{i,j}$ to video decoder **300**.

At video decoder **300**, the above calculations are reversed to produce $Q(r_{i,j})$, $0 \le i \le M-1$, $0 \le j \le N-1$. For vertical prediction case, it would be equation (3) below):

$$Q(r_{i,j}) = \Sigma_{k=0}^{i} \tilde{r}_{k,j}, \ 0 \le i \le (M-1), \ 0 \le j \le (N-1) \quad (3)$$

For horizontal case, it would be equation (4) below:

$$Q(r_{i,j}) = \Sigma_{k=0}^{j} \tilde{r}_{i,k}, \ 0 \le i \le (M-1), \ 0 \le j \le (N-1) \quad (4)$$

Video decoder **300** may add the inverse-quantized quantized residuals, $Q^{-1}(Q(r_{i,j}))$, to the original prediction values to produce reconstructed sample values.

The techniques described in the previous section may require compensation of coded transform coefficients with their predictors to derive the coefficient level to be dequantized. This can be done after an entire block is parsed. In other words, video decoder **300** may compensate the coded transform coefficients with their predictors after video decoder **300** parses the entire block. If it is required to perform the compensation operation during parsing (i.e., on the fly) and if a separate buffer is not used for storage of $Q(r_{i,j})$ values in addition to the $\tilde{r}_{i,j}$ values that are parsed, then the dependency of context derivation on neighboring coefficient groups (CGs) (i.e., across CGs) can be disabled for coefficient coding in transform skip mode for various syntax elements representing coefficient values. CGs are M×N groups of non-overlapping sets of transform coefficients within a coefficient block. If a part of a context template depends on a value across its own CG, then those values would be marked unavailable and the context would be derived that way. A context template is a spatial neighborhood from which a video coder gathers information to determine a coding context. For instance, in an example where a video coder uses a sum of absolute values of transform coefficients to determine a coding context, the context template defines the positions of such transform coefficients, e.g., the context template may define an above neighbor sample and a left neighbor sample as the samples to use for determining the coding context. This method allows overwriting of parsed coefficient values with their compensated (predict and add parsed coefficient) values to be inverse quantized in the decoder. Thus, it may not be necessary for video decoder **300** to use a separate buffer to store the compensated coefficient values. In some examples, the parsed transform coefficients can be overwritten by their

compensated values, and the contexts for future transform coefficients would use the overwritten values.

This disclosure may generally refer to "signaling" certain information, such as syntax elements. The term "signaling" may generally refer to the communication of values for syntax elements and/or other data used to decode encoded video data. That is, video encoder **200** may signal values for syntax elements in the bitstream. In general, signaling refers to generating a value in the bitstream. As noted above, source device **102** may transport the bitstream to destination device **116** substantially in real time, or not in real time, such as might occur when storing syntax elements to storage device **112** for later retrieval by destination device **116**.

FIGS. **2**A and **2**B are conceptual diagram illustrating an example quadtree binary tree (QTBT) structure **130**, and a corresponding coding tree unit (CTU) **132**. The solid lines represent quadtree splitting, and dotted lines indicate binary tree splitting. In each split (i.e., non-leaf) node of the binary tree, one flag is signaled to indicate which splitting type (i.e., horizontal or vertical) is used, where 0 indicates horizontal splitting and 1 indicates vertical splitting in this example. For the quadtree splitting, there is no need to indicate the splitting type, since quadtree nodes split a block horizontally and vertically into 4 sub-blocks with equal size. Accordingly, video encoder **200** may encode, and video decoder **300** may decode, syntax elements (such as splitting information) for a region tree level (i.e., the first level) of QTBT structure **130** (i.e., the solid lines) and syntax elements (such as splitting information) for a prediction tree level (i.e., the second level) of QTBT structure **130** (i.e., the dashed lines). Video encoder **200** may encode, and video decoder **300** may decode, video data, such as prediction and transform data, for CUs represented by terminal leaf nodes of QTBT structure **130**.

In general, CTU **132** of FIG. **2**B may be associated with parameters defining sizes of blocks corresponding to nodes of QTBT structure **130** at the first and second levels. These parameters may include a CTU size (representing a size of CTU **132** in samples), a minimum quadtree size (MinQTSize, representing a minimum allowed quadtree leaf node size), a maximum binary tree size (MaxBTSize, representing a maximum allowed binary tree root node size), a maximum binary tree depth (MaxBTDepth, representing a maximum allowed binary tree depth), and a minimum binary tree size (MinBTSize, representing the minimum allowed binary tree leaf node size).

The root node of a QTBT structure corresponding to a CTU may have four child nodes at the first level of the QTBT structure, each of which may be partitioned according to quadtree partitioning. That is, nodes of the first level are either leaf nodes (having no child nodes) or have four child nodes. The example of QTBT structure **130** represents such nodes as including the parent node and child nodes having solid lines for branches. If nodes of the first level are not larger than the maximum allowed binary tree root node size (MaxBTSize), they can be further partitioned by respective binary trees. The binary tree splitting of one node can be iterated until the nodes resulting from the split reach the minimum allowed binary tree leaf node size (MinBTSize) or the maximum allowed binary tree depth (MaxBTDepth). The example of QTBT structure **130** represents such nodes as having dashed lines for branches. The binary tree leaf node is referred to as a coding unit (CU), which is used for prediction (e.g., intra-picture or inter-picture prediction) and transform, without any further partitioning. As discussed above, CUs may also be referred to as "video blocks" or "blocks."

In one example of the QTBT partitioning structure, the CTU size is set as 128×128 (luma samples and two corresponding 64×64 chroma samples), the MinQTSize is set as 16×16, the MaxBTSize is set as 64×64, the MinBTSize (for both width and height) is set as 4, and the MaxBTDepth is set as 4. The quadtree partitioning is applied to the CTU first to generate quad-tree leaf nodes. The quadtree leaf nodes may have a size from 16×16 (i.e., the MinQTSize) to 128×128 (i.e., the CTU size). If the quadtree leaf node is 128×128, it will not be further split by the binary tree, since the size exceeds the MaxBTSize (i.e., 64×64, in this example). Otherwise, the quadtree leaf node will be further partitioned by the binary tree. Therefore, the quadtree leaf node is also the root node for the binary tree and has the binary tree depth as 0. When the binary tree depth reaches MaxBTDepth (4, in this example), no further splitting is permitted. When the binary tree node has width equal to MinBTSize (4, in this example), it implies that no further vertical splitting is permitted. Similarly, a binary tree node having a height equal to MinBTSize implies that no further horizontal splitting is permitted for that binary tree node. As noted above, leaf nodes of the binary tree are referred to as CUs and are further processed according to prediction and transform without further partitioning.

FIG. **3** is a block diagram illustrating an example video encoder **200** that may perform the techniques of this disclosure. FIG. **3** is provided for purposes of explanation and should not be considered limiting of the techniques as broadly exemplified and described in this disclosure. For purposes of explanation, this disclosure describes video encoder **200** in the context of video coding standards such as the HEVC video coding standard and the H.266 (VVC) video coding standard in development. However, the techniques of this disclosure are not limited to these video coding standards and are applicable generally to video encoding and decoding.

In the example of FIG. **3**, video encoder **200** includes video data memory **230**, mode selection unit **202**, residual generation unit **204**, transform processing unit **206**, quantization unit **208**, inverse quantization unit **210**, inverse transform processing unit **212**, reconstruction unit **214**, filter unit **216**, decoded picture buffer (DPB) **218**, and entropy encoding unit **220**. Any or all of video data memory **230**, mode selection unit **202**, residual generation unit **204**, transform processing unit **206**, quantization unit **208**, inverse quantization unit **210**, inverse transform processing unit **212**, reconstruction unit **214**, filter unit **216**, DPB **218**, and entropy encoding unit **220** may be implemented in one or more processors or in processing circuitry. Moreover, video encoder **200** may include additional or alternative processors or processing circuitry to perform these and other functions.

Video data memory **230** may store video data to be encoded by the components of video encoder **200**. Video encoder **200** may receive the video data stored in video data memory **230** from, for example, video source **104** (FIG. **1**). DPB **218** may act as a reference picture memory that stores reference video data for use in prediction of subsequent video data by video encoder **200**. Video data memory **230** and DPB **218** may be formed by any of a variety of memory devices, such as dynamic random access memory (DRAM), including synchronous DRAM (SDRAM), magnetoresistive RAM (MRAM), resistive RAM (RRAM), or other types of memory devices. Video data memory **230** and DPB **218** may be provided by the same memory device or separate memory devices. In various examples, video data memory **230** may be on-chip with other components of video encoder **200**, as illustrated, or off-chip relative to those components.

In this disclosure, reference to video data memory **230** should not be interpreted as being limited to memory internal to video encoder **200**, unless specifically described as such, or memory external to video encoder **200**, unless specifically described as such. Rather, reference to video data memory **230** should be understood as reference memory that stores video data that video encoder **200** receives for encoding (e.g., video data for a current block that is to be encoded). Memory **106** of FIG. **1** may also provide temporary storage of outputs from the various units of video encoder **200**.

The various units of FIG. **3** are illustrated to assist with understanding the operations performed by video encoder **200**. The units may be implemented as fixed-function circuits, programmable circuits, or a combination thereof. Fixed-function circuits refer to circuits that provide particular functionality and are preset on the operations that can be performed. Programmable circuits refer to circuits that can be programmed to perform various tasks and provide flexible functionality in the operations that can be performed. For instance, programmable circuits may execute software or firmware that cause the programmable circuits to operate in the manner defined by instructions of the software or firmware. Fixed-function circuits may execute software instructions (e.g., to receive parameters or output parameters), but the types of operations that the fixed-function circuits perform are generally immutable. In some examples, one or more of the units may be distinct circuit blocks (fixed-function or programmable), and in some examples, the one or more units may be integrated circuits.

Video encoder **200** may include arithmetic logic units (ALUs), elementary function units (EFUs), digital circuits, analog circuits, and/or programmable cores, formed from programmable circuits. In examples where the operations of video encoder **200** are performed using software executed by the programmable circuits, memory **106** (FIG. **1**) may store the object code of the software that video encoder **200** receives and executes, or another memory within video encoder **200** (not shown) may store such instructions.

Video data memory **230** is configured to store received video data. Video encoder **200** may retrieve a picture of the video data from video data memory **230** and provide the video data to residual generation unit **204** and mode selection unit **202**. Video data in video data memory **230** may be raw video data that is to be encoded.

Mode selection unit **202** includes a motion estimation unit **222**, motion compensation unit **224**, and an intra-prediction unit **226**. Mode selection unit **202** may include additional functional units to perform video prediction in accordance with other prediction modes. As examples, mode selection unit **202** may include a palette unit, an intra-block copy unit (which may be part of motion estimation unit **222** and/or motion compensation unit **224**), an affine unit, a linear model (LM) unit, or the like.

Mode selection unit **202** generally coordinates multiple encoding passes to test combinations of encoding parameters and resulting rate-distortion values for such combinations. The encoding parameters may include partitioning of CTUs into CUs, prediction modes for the CUs, transform types for residual data of the CUs, quantization parameters for residual data of the CUs, and so on. Mode selection unit **202** may ultimately select the combination of encoding parameters having rate-distortion values that are better than the other tested combinations.

Video encoder **200** may partition a picture retrieved from video data memory **230** into a series of CTUs and encapsulate one or more CTUs within a slice. Mode selection unit **202** may partition a CTU of the picture in accordance with a tree structure, such as the QTBT structure or the quad-tree structure of HEVC described above. As described above, video encoder **200** may form one or more CUs from partitioning a CTU according to the tree structure. Such a CU may also be referred to generally as a "video block" or "block."

In general, mode selection unit **202** also controls the components thereof (e.g., motion estimation unit **222**, motion compensation unit **224**, and intra-prediction unit **226**) to generate a prediction block for a current block (e.g., a current CU, or in HEVC, the overlapping portion of a PU and a TU). For inter-prediction of a current block, motion estimation unit **222** may perform a motion search to identify one or more closely matching reference blocks in one or more reference pictures (e.g., one or more previously coded pictures stored in DPB **218**). In particular, motion estimation unit **222** may calculate a value representative of how similar a potential reference block is to the current block, e.g., according to sum of absolute difference (SAD), sum of squared differences (SSD), mean absolute difference (MAD), mean squared differences (MSD), or the like. Motion estimation unit **222** may generally perform these calculations using sample-by-sample differences between the current block and the reference block being considered. Motion estimation unit **222** may identify a reference block having a lowest value resulting from these calculations, indicating a reference block that most closely matches the current block.

Motion estimation unit **222** may form one or more motion vectors (MVs) that defines the positions of the reference blocks in the reference pictures relative to the position of the current block in a current picture. Motion estimation unit **222** may then provide the motion vectors to motion compensation unit **224**. For example, for uni-directional inter-prediction, motion estimation unit **222** may provide a single motion vector, whereas for bi-directional inter-prediction, motion estimation unit **222** may provide two motion vectors. Motion compensation unit **224** may then generate a prediction block using the motion vectors. For example, motion compensation unit **224** may retrieve data of the reference block using the motion vector. As another example, if the motion vector has fractional sample precision, motion compensation unit **224** may interpolate values for the prediction block according to one or more interpolation filters. Moreover, for bi-directional inter-prediction, motion compensation unit **224** may retrieve data for two reference blocks identified by respective motion vectors and combine the retrieved data, e.g., through sample-by-sample averaging or weighted averaging.

As another example, for intra-prediction, or intra-prediction coding, intra-prediction unit **226** may generate the prediction block from samples neighboring the current block. For example, for directional modes, intra-prediction unit **226** may generally mathematically combine values of neighboring samples and populate these calculated values in the defined direction across the current block to produce the prediction block. As another example, for DC mode, intra-prediction unit **226** may calculate an average of the neighboring samples to the current block and generate the prediction block to include this resulting average for each sample of the prediction block.

Mode selection unit **202** provides the prediction block to residual generation unit **204**. Residual generation unit **204** receives a raw, uncoded version of the current block from video data memory **230** and the prediction block from mode selection unit **202**. Residual generation unit **204** calculates

sample-by-sample differences between the current block and the prediction block. The resulting sample-by-sample differences define a residual block for the current block. In some examples, residual generation unit **204** may also determine differences between sample values in the residual block to generate a residual block using residual differential pulse code modulation (RDPCM). In some examples, residual generation unit **204** may be formed using one or more subtractor circuits that perform binary subtraction.

In examples where mode selection unit **202** partitions CUs into PUs, each PU may be associated with a luma prediction unit and corresponding chroma prediction units. Video encoder **200** and video decoder **300** may support PUs having various sizes. As indicated above, the size of a CU may refer to the size of the luma coding block of the CU and the size of a PU may refer to the size of a luma prediction unit of the PU. Assuming that the size of a particular CU is 2N×2N, video encoder **200** may support PU sizes of 2N×2N or N×N for intra prediction, and symmetric PU sizes of 2N×2N, 2N×N, N×2N, N×N, or similar for inter prediction. Video encoder **200** and video decoder **300** may also support asymmetric partitioning for PU sizes of 2N×nU, 2N×nD, nL×2N, and nR×2N for inter prediction.

In examples where mode selection unit does not further partition a CU into PUs, each CU may be associated with a luma coding block and corresponding chroma coding blocks. As above, the size of a CU may refer to the size of the luma coding block of the CU. The video encoder **200** and video decoder **300** may support CU sizes of 2N×2N, 2N×N, or N×2N.

For other video coding techniques such as intra-block copy mode coding, affine-mode coding, and linear model (LM) mode coding, as few as examples, mode selection unit **202**, via respective units associated with the coding techniques, generates a prediction block for the current block being encoded. In some examples, such as palette mode coding, mode selection unit **202** may not generate a prediction block, and instead generate syntax elements that indicate the manner in which to reconstruct the block based on a selected palette. In such modes, mode selection unit **202** may provide these syntax elements to entropy encoding unit **220** to be encoded.

As described above, residual generation unit **204** receives the video data for the current block and the corresponding prediction block. Residual generation unit **204** then generates a residual block for the current block. To generate the residual block, residual generation unit **204** calculates sample-by-sample differences between the prediction block and the current block.

Transform processing unit **206** applies one or more transforms to the residual block to generate a block of transform coefficients (referred to herein as a "transform coefficient block"). Transform processing unit **206** may apply various transforms to a residual block to form the transform coefficient block. For example, transform processing unit **206** may apply a discrete cosine transform (DCT), a directional transform, a Karhunen-Loeve transform (KLT), or a conceptually similar transform to a residual block. In some examples, transform processing unit **206** may perform multiple transforms to a residual block, e.g., a primary transform and a secondary transform, such as a rotational transform. In some examples, transform processing unit **206** does not apply transforms to a residual block. For instance, in examples where a block is coded using coefficient-domain BDPCM, transform processing unit **206** may skip application of the transform to the residual values generated by residual generation unit **204**.

Quantization unit **208** may quantize the transform coefficients in a transform coefficient block, to produce a quantized transform coefficient block. Quantization unit **208** may quantize transform coefficients of a transform coefficient block according to a quantization parameter (QP) value associated with the current block. Video encoder **200** (e.g., via mode selection unit **202**) may adjust the degree of quantization applied to the coefficient blocks associated with the current block by adjusting the QP value associated with the CU. Quantization may introduce loss of information, and thus, quantized transform coefficients may have lower precision than the original transform coefficients produced by transform processing unit **206**.

In accordance with one or more techniques of this disclosure, quantization unit **208** may use coefficient-domain BDPCM to determine residual quantized samples based on the quantized residual values. For example, quantization unit **208** may use equations (1) or (2), above, to determine the residual quantized samples. Quantization unit **208** may determine whether intra-prediction unit **226** used a vertical intra prediction mode or a horizontal intra prediction mode to generate the prediction block that residual generation unit **204** used to generate the residual values. If intra-prediction unit **226** used the vertical intra prediction mode (i.e., the intra prediction being vertical prediction), quantization unit **208** may use equation (1). If intra-prediction unit **226** used the horizontal intra prediction mode (i.e., the intra prediction being horizontal prediction), quantization unit **208** may use equation (2). For example, quantization unit **208** may determine whether vertical prediction or horizontal prediction yields greater coding efficiency.

Inverse quantization unit **210** and inverse transform processing unit **212** may apply inverse quantization and inverse transforms to a quantized transform coefficient block, respectively, to reconstruct a residual block from the transform coefficient block. In accordance with one or more examples of this disclosure, if a block is coded using coefficient-domain BDPCM, inverse quantization unit **210** may determine quantized residual values based on the residual quantized samples, e.g., by applying equation (3) or equation (4). More specifically, if the block was coded using a vertical intra prediction mode, inverse quantization unit **210** may apply equation (3). If the block was coded using a horizontal intra prediction mode, inverse quantization unit **210** may apply equation (4). After determining the quantized residual values, inverse quantization unit **210** may inverse quantize the quantized residual values.

After inverse quantization unit **210** forms the transform coefficient block, inverse transform processing unit **212** may apply one or more inverse transforms to the transform coefficient block to generate a residual block associated with the current block. For example, inverse transform processing unit **212** may apply an inverse DCT, an inverse integer transform, an inverse Karhunen-Loeve transform (KLT), an inverse rotational transform, an inverse directional transform, or another inverse transform to the coefficient block. In examples where a block is coded using coefficient-domain BDPCM is used, inverse transform processing unit **212** may skip application of the inverse transform.

Reconstruction unit **214** may produce a reconstructed block corresponding to the current block (albeit potentially with some degree of distortion) based on the reconstructed residual block and a prediction block generated by mode selection unit **202**. For example, reconstruction unit **214** may add samples of the reconstructed residual block to corresponding samples from the prediction block generated by mode selection unit **202** to produce the reconstructed block.

Filter unit **216** may perform one or more filter operations on reconstructed blocks. For example, filter unit **216** may perform deblocking operations to reduce blockiness artifacts along edges of CUs. Operations of filter unit **216** may be skipped, in some examples.

Video encoder **200** stores reconstructed blocks in DPB **218**. For instance, in examples where operations of filter unit **216** are not needed, reconstruction unit **214** may store reconstructed blocks to DPB **218**. In examples where operations of filter unit **216** are needed, filter unit **216** may store the filtered reconstructed blocks to DPB **218**. Motion estimation unit **222** and motion compensation unit **224** may retrieve a reference picture from DPB **218**, formed from the reconstructed (and potentially filtered) blocks, to inter-predict blocks of subsequently encoded pictures. In addition, intra-prediction unit **226** may use reconstructed blocks in DPB **218** of a current picture to intra-predict other blocks in the current picture.

In general, entropy encoding unit **220** may entropy encode syntax elements received from other functional components of video encoder **200**. For example, entropy encoding unit **220** may entropy encode quantized transform coefficient blocks from quantization unit **208**. As another example, entropy encoding unit **220** may entropy encode prediction syntax elements (e.g., motion information for inter-prediction or intra-mode information for intra-prediction) from mode selection unit **202**. Entropy encoding unit **220** may perform one or more entropy encoding operations on the syntax elements, which are another example of video data, to generate entropy-encoded data. For example, entropy encoding unit **220** may perform a context-adaptive variable length coding (CAVLC) operation, a CABAC operation, a variable-to-variable (VV) length coding operation, a syntax-based context-adaptive binary arithmetic coding (SBAC) operation, a Probability Interval Partitioning Entropy (PIPE) coding operation, an Exponential-Golomb encoding operation, or another type of entropy encoding operation on the data. In some examples, entropy encoding unit **220** may operate in bypass mode where syntax elements are not entropy encoded.

Video encoder **200** may output a bitstream that includes the entropy encoded syntax elements needed to reconstruct blocks of a slice or picture. For instance, in the example of FIG. **3**, entropy encoding unit **220** may output the bitstream. In accordance with one or more examples of this disclosure, video encoder **200** may signal residual quantized samples in the bitstream. For instance, entropy encoding unit **220** may entropy encode syntax elements representing residual quantized samples and include the entropy-encoded syntax elements in the bitstream.

The operations described above are described with respect to a block. Such description should be understood as being operations for a luma coding block and/or chroma coding blocks. As described above, in some examples, the luma coding block and chroma coding blocks are luma and chroma components of a CU. In some examples, the luma coding block and the chroma coding blocks are luma and chroma components of a PU.

In some examples, operations performed with respect to a luma coding block need not be repeated for the chroma coding blocks. As one example, operations to identify a motion vector (MV) and reference picture for a luma coding block need not be repeated for identifying an MV and reference picture for the chroma blocks. Rather, the MV for the luma coding block may be scaled to determine the MV for the chroma blocks, and the reference picture may be the same. As another example, the intra-prediction process may be the same for the luma coding blocks and the chroma coding blocks.

Video encoder **200** represents an example of a device configured to encode video data including a memory configured to store video data, and one or more processing units implemented in circuitry and configured to generate predicted values by performing intra prediction for a block of the video data using unfiltered samples from above or left block boundary samples; generate residual values based on original sample values of the block and the predicted values; quantize the residual values; and determine residual quantized samples based on the quantized residual values.

FIG. **4** is a block diagram illustrating an example video decoder **300** that may perform the techniques of this disclosure. FIG. **4** is provided for purposes of explanation and is not limiting on the techniques as broadly exemplified and described in this disclosure. For purposes of explanation, this disclosure describes video decoder **300** according to the techniques of JEM, VVC, and HEVC. However, the techniques of this disclosure may be performed by video coding devices that are configured to other video coding standards.

In the example of FIG. **4**, video decoder **300** includes coded picture buffer (CPB) memory **320**, entropy decoding unit **302**, prediction processing unit **304**, inverse quantization unit **306**, inverse transform processing unit **308**, reconstruction unit **310**, filter unit **312**, and decoded picture buffer (DPB) **314**. Any or all of CPB memory **320**, entropy decoding unit **302**, prediction processing unit **304**, inverse quantization unit **306**, inverse transform processing unit **308**, reconstruction unit **310**, filter unit **312**, and DPB **314** may be implemented in one or more processors or in processing circuitry. Moreover, video decoder **300** may include additional or alternative processors or processing circuitry to perform these and other functions.

Prediction processing unit **304** includes motion compensation unit **316** and intra-prediction unit **318**. Prediction processing unit **304** may include additional units to perform prediction in accordance with other prediction modes. As examples, prediction processing unit **304** may include a palette unit, an intra-block copy unit (which may form part of motion compensation unit **316**), an affine unit, a linear model (LM) unit, or the like. In other examples, video decoder **300** may include more, fewer, or different functional components.

CPB memory **320** may store video data, such as an encoded video bitstream, to be decoded by the components of video decoder **300**. The video data stored in CPB memory **320** may be obtained, for example, from computer-readable medium **110** (FIG. **1**). CPB memory **320** may include a CPB that stores encoded video data (e.g., syntax elements) from an encoded video bitstream. Also, CPB memory **320** may store video data other than syntax elements of a coded picture, such as temporary data representing outputs from the various units of video decoder **300**. DPB **314** generally stores decoded pictures, which video decoder **300** may output and/or use as reference video data when decoding subsequent data or pictures of the encoded video bitstream. CPB memory **320** and DPB **314** may be formed by any of a variety of memory devices, such as dynamic random access memory (DRAM), including synchronous DRAM (SDRAM), magnetoresistive RAM (MRAM), resistive RAM (RRAM), or other types of memory devices. CPB memory **320** and DPB **314** may be provided by the same memory device or separate memory devices. In various

examples, CPB memory **320** may be on-chip with other components of video decoder **300**, or off-chip relative to those components.

Additionally or alternatively, in some examples, video decoder **300** may retrieve coded video data from memory **120** (FIG. **1**). That is, memory **120** may store data as discussed above with CPB memory **320**. Likewise, memory **120** may store instructions to be executed by video decoder **300**, when some or all of the functionality of video decoder **300** is implemented in software to be executed by processing circuitry of video decoder **300**.

The various units shown in FIG. **4** are illustrated to assist with understanding the operations performed by video decoder **300**. The units may be implemented as fixed-function circuits, programmable circuits, or a combination thereof. Similar to FIG. **3**, fixed-function circuits refer to circuits that provide particular functionality, and are preset on the operations that can be performed. Programmable circuits refer to circuits that can be programmed to perform various tasks and provide flexible functionality in the operations that can be performed. For instance, programmable circuits may execute software or firmware that cause the programmable circuits to operate in the manner defined by instructions of the software or firmware. Fixed-function circuits may execute software instructions (e.g., to receive parameters or output parameters), but the types of operations that the fixed-function circuits perform are generally immutable. In some examples, one or more of the units may be distinct circuit blocks (fixed-function or programmable), and in some examples, the one or more units may be integrated circuits.

Video decoder **300** may include ALUs, EFUs, digital circuits, analog circuits, and/or programmable cores formed from programmable circuits. In examples where the operations of video decoder **300** are performed by software executing on the programmable circuits, on-chip or off-chip memory may store instructions (e.g., object code) of the software that video decoder **300** receives and executes.

Entropy decoding unit **302** may receive encoded video data from the CPB and entropy decode the video data to reproduce syntax elements. Prediction processing unit **304**, inverse quantization unit **306**, inverse transform processing unit **308**, reconstruction unit **310**, and filter unit **312** may generate decoded video data based on the syntax elements extracted from the bitstream.

In general, video decoder **300** reconstructs a picture on a block-by-block basis. Video decoder **300** may perform a reconstruction operation on each block individually (where the block currently being reconstructed, i.e., decoded, may be referred to as a "current block").

Entropy decoding unit **302** may entropy decode syntax elements defining quantized transform coefficients of a quantized transform coefficient block, as well as transform information, such as a quantization parameter (QP) and/or transform mode indication(s). Inverse quantization unit **306** may use the QP associated with the quantized transform coefficient block to determine a degree of quantization and, likewise, a degree of inverse quantization for inverse quantization unit **306** to apply. Inverse quantization unit **306** may, for example, perform a bitwise left-shift operation to inverse quantize the quantized transform coefficients. Inverse quantization unit **306** may thereby form a transform coefficient block including transform coefficients.

In accordance with one or more techniques of this disclosure, inverse quantization unit **306** may determine, based on syntax elements obtained by entropy decoding unit **302** from a bitstream, residual quantized samples of a block of the video data. Additionally, inverse quantization unit **306** may determine quantized residual values based on the residual quantized samples. For instance, inverse quantization unit **306** may apply equations (3) or (4) to determine the quantized residual values. More specifically, if the block was coded using a vertical intra prediction mode, inverse quantization unit 306 may apply equation (3). If the block was coded using a horizontal intra prediction mode, inverse quantization unit 306 may apply equation (4). After determining the quantized residual values, inverse quantization unit **306** may inverse quantize the quantized residual values.

After inverse quantization unit **306** forms the transform coefficient block, inverse transform processing unit **308** may apply one or more inverse transforms to the transform coefficient block to generate a residual block associated with the current block. For example, inverse transform processing unit **308** may apply an inverse DCT, an inverse integer transform, an inverse Karhunen-Loeve transform (KLT), an inverse rotational transform, an inverse directional transform, or another inverse transform to the transform coefficient block. In examples where a block is coded using coefficient-domain BDPCM is used, inverse transform processing unit **308** may skip application of the inverse transform.

Furthermore, prediction processing unit **304** generates a prediction block according to prediction information syntax elements that were entropy decoded by entropy decoding unit **302**. For example, if the prediction information syntax elements indicate that the current block is inter-predicted, motion compensation unit **316** may generate the prediction block. In this case, the prediction information syntax elements may indicate a reference picture in DPB **314** from which to retrieve a reference block, as well as a motion vector identifying a location of the reference block in the reference picture relative to the location of the current block in the current picture. Motion compensation unit **316** may generally perform the inter-prediction process in a manner that is substantially similar to that described with respect to motion compensation unit **224** (FIG. **3**).

As another example, if the prediction information syntax elements indicate that the current block is intra-predicted, intra-prediction unit **318** may generate the prediction block according to an intra-prediction mode indicated by the prediction information syntax elements. Again, intra-prediction unit **318** may generally perform the intra-prediction process in a manner that is substantially similar to that described with respect to intra-prediction unit **226** (FIG. **3**). Intra-prediction unit **318** may retrieve data of neighboring samples to the current block from DPB **314**.

Reconstruction unit **310** may reconstruct the current block using the prediction block and the residual block. For example, reconstruction unit **310** may add samples of the residual block to corresponding samples of the prediction block to reconstruct the current block.

Filter unit **312** may perform one or more filter operations on reconstructed blocks. For example, filter unit **312** may perform deblocking operations to reduce blockiness artifacts along edges of the reconstructed blocks. Operations of filter unit **312** are not necessarily performed in all examples.

Video decoder **300** may store the reconstructed blocks in DPB **314**. For instance, in examples where operations of filter unit **312** are not performed, reconstruction unit **310** may store reconstructed blocks to DPB **314**. In examples where operations of filter unit **312** are performed, filter unit **312** may store the filtered reconstructed blocks to DPB **314**. As discussed above, DPB **314** may provide reference information, such as samples of a current picture for intra-

prediction and previously decoded pictures for subsequent motion compensation, to prediction processing unit **304**. Moreover, video decoder **300** may output decoded pictures from DPB for subsequent presentation on a display device, such as display device **118** of FIG. **1**.

In this manner, video decoder **300** represents an example of a video decoding device including a memory configured to store video data, and one or more processing units implemented in circuitry and configured to determine, based on syntax elements in a bitstream that comprises an encoded representation of the video data, residual quantized samples of a block of the video data; determine quantized residual values based on the residual quantized samples; inverse quantize the quantized residual values; generate predicted values by performing intra prediction for the block using unfiltered samples from above or left block boundary samples; and reconstruct original sample values of the block based on the inverse-quantized quantized residuals and the prediction values.

In the example of FIG. **4**, entropy decoding unit **302** of video decoder **300** may determine, based on syntax elements in a bitstream that comprises an encoded representation of the video data, residual quantized samples of a block of the video data. Furthermore, in the example of FIG. **4**, video decoder **300** may determine quantized residual values based on the residual quantized samples; inverse quantize the quantized residual values; generate predicted values by performing intra prediction for the block using unfiltered samples from above or left block boundary samples; and reconstruct original sample values of the block based on the inverse-quantized quantized residuals and the prediction values.

FIG. **5** is a flowchart illustrating an example method for encoding a current block. The current block may comprise a current CU. Although described with respect to video encoder **200** (FIGS. **1** and **2**), it should be understood that other devices may be configured to perform a method similar to that of FIG. **5**.

In this example, video encoder **200** initially predicts the current block (**350**). For example, video encoder **200** may form a prediction block for the current block. Video encoder **200** may then calculate a residual block for the current block (**352**). To calculate the residual block, video encoder **200** may calculate a difference between the original, uncoded block and the prediction block for the current block. Video encoder **200** may then transform difference values to produce transform coefficients and quantize the transform coefficients of the residual block (**354**). In accordance with the techniques of this disclosure, video encoder **200** may skip application of the transform and may perform coefficient-domain BDPCM using the quantized residual values. Next, video encoder **200** may scan the quantized transform coefficients (or quantized residual values when video encoder **200** applies coefficient-domain BDPCM) of the residual block (**356**). During the scan, or following the scan, video encoder **200** may entropy encode the transform coefficients (or quantized residual values when video encoder **200** applies coefficient-domain BDPCM) (**358**). For example, video encoder **200** may encode the transform coefficients (or quantized residual values when video encoder **200** applies coefficient-domain BDPCM) using CAVLC or CABAC. Video encoder **200** may then output the entropy encoded data of the block (**360**). Thus, in examples where video encoder **200** applies coefficient-domain BDPCM, video encoder **200** may signal the residual quantized values.

FIG. **6** is a flowchart illustrating an example method for decoding a current block of video data. The current block may comprise a current CU. Although described with respect to video decoder **300** (FIGS. **1** and **3**), it should be understood that other devices may be configured to perform a method similar to that of FIG. **6**.

Video decoder **300** may receive entropy coded data for the current block, such as entropy coded prediction information and entropy coded data for transform coefficients of a residual block corresponding to the current block (**370**). Video decoder **300** may entropy decode the entropy coded data to determine prediction information for the current block and to reproduce transform coefficients of the residual block (**372**). Video decoder **300** may predict the current block (**374**), e.g., using an intra- or inter-prediction mode as indicated by the prediction information for the current block, to calculate a prediction block for the current block. Video decoder **300** may then inverse scan the reproduced transform coefficients (**376**), to create a block of quantized transform coefficients. Video decoder **300** may then inverse quantize and apply and inverse transform to the transform coefficients to produce a residual block (**378**). Video decoder **300** may perform the techniques of this disclosure for coefficient-domain BDPCM as part of the step of producing the residual block. Video decoder **300** may ultimately decode the current block by combining the prediction block and the residual block (**380**).

FIG. **7** is a flowchart illustrating an example video encoding process that includes coefficient domain block-differential pulse-code modulation (BDPCM), in accordance with one or more techniques of this disclosure. FIG. **7** may be a more specific instance of the operation of FIG. **5** in which video encoder **200** uses coefficient-domain BDPCM. In the example of FIG. **7**, video encoder **200** may generate a block of predicted values by performing intra prediction for a block of the video data using unfiltered samples from above or left block boundary samples (**700**). For instance, video encoder **200** may perform intra prediction in accordance with any of the examples provided elsewhere in this disclosure.

Additionally, video encoder **200** may generate residual values based on original sample values of the block and the predicted values (**702**). Each of the residual values may indicate a difference between one of the original sample values of the block and a corresponding predicted value.

Furthermore, in the example of FIG. **7**, video encoder **200** may quantize the residual values (**704**). For example, video encoder **200** may quantize the residual values in accordance with any of the examples for quantizing described elsewhere in this disclosure.

After quantizing the residual values, video encoder **200** may determine residual quantized samples based on the quantized residual values (**706**). In examples where the intra prediction is vertical prediction, video encoder **200** may determine the residual quantized samples as:

$$\tilde{r}_{i,j} = \begin{cases} Q(r_{i,j}), & i = 0,\ 0 \le j \le (N-1) \\ Q(r_{i,j}) - Q(r_{(i-1),j}), & 1 \le i \le (M-1),\ 0 \le j \le (N-1) \end{cases}$$

where $\tilde{r}_{k,j}$ is a residual quantized sample at position i,j, $Q(r_{i,j})$ and $Q(r_{(i-1),j})$ are quantized residual values, M is a number of rows of the block, and N is a number of columns of the block. In examples where intra prediction is horizontal prediction, video encoder **200** may determine the residual quantized samples as:

$$\tilde{r}_{i,j} = \begin{cases} Q(r_{i,j}), & 0 \le i \le (M-1),\ j = 0 \\ Q(r_{i,j}) - Q(r_{i,(j-1)}), & 0 \le i \le (M-1),\ 1 \le j \le (N-1) \end{cases}$$

where $\tilde{r}_{k,j}$ is a residual quantized sample at position i,j, $Q(r_{i,j})$ and $Q(r_{(i-1),j})$ are quantized residual values, M is a number of rows of the block, and N is a number of columns of the block.

Video encoder **200** may signal the residual quantized samples (**708**). For example, video encoder **200** may include entropy-encoded syntax elements represent the residual quantized samples in a bitstream.

FIG. **8** is a flowchart illustrating an example video decoding process that includes coefficient domain BDPCM, in accordance with one or more techniques of this disclosure. FIG. **8** may be a more specific instance of the operation of FIG. **5** in which video encoder **200** uses coefficient-domain BDPCM. In the example of FIG. **8**, video decoder **300** may determine, based on syntax elements in a bitstream that comprises an encoded representation of the video data, residual quantized samples of a block of the video data (**800**). For example, video decoder **300** may use entropy decoding to decode syntax elements in a bitstream that indicate the residual quantized samples of the block.

Furthermore, in the example of FIG. **8**, video decoder **300** may determine quantized residual values based on the residual quantized samples (**802**). In examples where the intra prediction is vertical prediction, video decoder **300** may determine the quantized residual values as:

$$Q(r_{i,j}) = \sum_{k=0}^{i} \tilde{r}_{k,j},\ 0 \le i \le (M-1),\ 0 \le j \le (N-1)$$

where $Q(r_{i,j})$ is a quantized residual value at position i,j, $\tilde{r}_{k,j}$ is one of the residual quantized samples, M is a number of rows of the block, and N is a number of columns of the block. In examples where the intra prediction is horizontal prediction, video decoder **300** may determine the quantized residual values as:

$$Q(r_{i,j}) = \sum_{k=0}^{j} \tilde{r}_{i,k},\ 0 \le i \le (M-1),\ 0 \le j \le (N-1)$$

where $Q(r_{i,j})$ is a quantized residual value at position i,j, $\tilde{r}_{k,j}$ is one of the residual quantized samples, M is a number of rows of the block, and N is a number of columns of the block.

After determining the quantized residual values, video decoder **300** may inverse quantize the quantized residual values (**804**). For example, video decoder **300** may inverse quantize the quantized residual values by inversing any of the examples of quantization provided elsewhere in this disclosure.

Video decoder **300** may generate predicted values by performing intra prediction for the block using unfiltered samples from above or left block boundary samples (**806**). For instance, video decoder **300** may perform intra prediction in accordance with any of the examples provided elsewhere in this disclosure.

Video decoder **300** may reconstruct original sample values of the block based on the inverse-quantized quantized residual values and the prediction values (**808**). For example, video decoder **300** may add inverse-quantized quantized residual values to corresponding prediction values to reconstruct the original sample values.

The following paragraphs provide a non-limiting list of enumerated examples in accordance with the techniques of this disclosure.

EXAMPLE 1

A method of decoding video data, the method comprising: determining, based on syntax elements in a bitstream that comprises an encoded representation of the video data, residual quantized samples of a block of the video data; determining quantized residual values based on the residual quantized samples; inverse quantizing the quantized residual values; generating predicted values by performing intra prediction for the block using unfiltered samples from above or left block boundary samples; and reconstructing original sample values of the block based on the inverse-quantized quantized residuals and the prediction values.

EXAMPLE 2

The method of example 1, wherein determining the quantized residual values comprises: based on the intra prediction being vertical prediction, determining the quantized residual values as:

$$Q(r_{i,j}) = \sum_{k=0}^{i} \tilde{r}_{k,j},\ 0 \le i \le (M-1),\ 0 \le j \le (N-1)$$

where $Q(r_{i,j})$ is a quantized residual value at position i,j, $\tilde{r}_{k,j}$ is one of the residual quantized samples, M is a number of rows of the block, and N is a number of columns of the block.

EXAMPLE 3

The method of example 1, wherein determining the quantized residual values comprises: based on the intra prediction being horizontal prediction, determining the quantized residual values as:

$$Q(r_{i,j}) = \sum_{k=0}^{j} \tilde{r}_{i,k},\ 0 \le i \le (M-1),\ 0 \le j \le (N-1)$$

where $Q(r_{i,j})$ is a quantized residual value at position i,j, $\tilde{r}_{k,j}$ is one of the residual quantized samples, M is a number of rows of the block, and N is a number of columns of the block.

EXAMPLE 4

A method of encoding video data, the method comprising: generating predicted values by performing intra prediction for a block of the video data using unfiltered samples from above or left block boundary samples; generating residual values based on original sample values of the block and the predicted values; quantizing the residual values; and determining residual quantized samples based on the quantized residual values.

EXAMPLE 5

The method of example 4, wherein determining the residual quantized samples comprises: based on the intra prediction being vertical prediction, determining the quantized residuals as:

$$\tilde{r}_{i,j} = \begin{cases} Q(r_{i,j}), & i = 0, \quad 0 \le j \le (N-1) \\ Q(r_{i,j}) - Q(r_{(i-1),j}), & 1 \le i \le (M-1), \quad 0 \le j \le (N-1) \end{cases}$$

where $\tilde{r}_{k,j}$ is a residual quantized sample at position i,j, $Q(r_{i,j})$ and $Q(r_{(i-1),j})$ are quantized residual values, M is a number of rows of the block, and N is a number of columns of the block.

EXAMPLE 6

The method of example 4, wherein determining the quantized residuals comprises: based on the intra prediction being horizontal prediction, determining the quantized residuals as:

$$\tilde{r}_{i,j} = \begin{cases} Q(r_{i,j}), & 0 \le i \le (M-1), \quad j = 0 \\ Q(r_{i,j}) - Q(r_{i,O-1)}), & 0 \le i \le (M-1), \quad 1 \le j \le (N-1) \end{cases}$$

where $\tilde{r}_{k,j}$ is a residual quantized sample at position i,j, $Q(r_{i,j})$ and $Q(r_{(i-1),j})$ are quantized residual values, M is a number of rows of the block, and N is a number of columns of the block.

EXAMPLE 7

A device for coding video data, the device comprising one or more means for performing the method of any of examples 1-6.

EXAMPLE 8

The device of example 7, wherein the one or more means comprise one or more processors implemented in circuitry.

EXAMPLE 9

The device of any of examples 7 and 8, further comprising a memory to store the video data.

EXAMPLE 10

The device of any of examples 7-9, further comprising a display configured to display decoded video data.

EXAMPLE 11

The device of any of examples 7-10, wherein the device comprises one or more of a camera, a computer, a mobile device, a broadcast receiver device, or a set-top box.

EXAMPLE 12

The device of any of examples 7-11, wherein the device comprises a video decoder.

EXAMPLE 13

The device of any of examples 7-12, wherein the device comprises a video encoder.

EXAMPLE 14

A computer-readable storage medium having stored thereon instructions that, when executed, cause one or more processors to perform the method of any of examples 1-6.

EXAMPLE 15

A device for encoding video data, the device comprising means for performing the methods of any of examples 1-6.

It is to be recognized that depending on the example, certain acts or events of any of the techniques described herein can be performed in a different sequence, may be added, merged, or left out altogether (e.g., not all described acts or events are necessary for the practice of the techniques). Moreover, in certain examples, acts or events may be performed concurrently, e.g., through multi-threaded processing, interrupt processing, or multiple processors, rather than sequentially.

In one or more examples, the functions described may be implemented in hardware, software, firmware, or any combination thereof. If implemented in software, the functions may be stored on or transmitted over as one or more instructions or code on a computer-readable medium and executed by a hardware-based processing unit. Computer-readable media may include computer-readable storage media, which corresponds to a tangible medium such as data storage media, or communication media including any medium that facilitates transfer of a computer program from one place to another, e.g., according to a communication protocol. In this manner, computer-readable media generally may correspond to (1) tangible computer-readable storage media which is non-transitory or (2) a communication medium such as a signal or carrier wave. Data storage media may be any available media that can be accessed by one or more computers or one or more processors to retrieve instructions, code and/or data structures for implementation of the techniques described in this disclosure. A computer program product may include a computer-readable medium.

By way of example, and not limitation, such computer-readable storage media can comprise RAM, ROM, EEPROM, CD-ROM or other optical disk storage, magnetic disk storage, or other magnetic storage devices, flash memory, or any other medium that can be used to store desired program code in the form of instructions or data structures and that can be accessed by a computer. Also, any connection is properly termed a computer-readable medium. For example, if instructions are transmitted from a website, server, or other remote source using a coaxial cable, fiber optic cable, twisted pair, digital subscriber line (DSL), or wireless technologies such as infrared, radio, and microwave, then the coaxial cable, fiber optic cable, twisted pair, DSL, or wireless technologies such as infrared, radio, and microwave are included in the definition of medium. It should be understood, however, that computer-readable storage media and data storage media do not include connections, carrier waves, signals, or other transitory media, but are instead directed to non-transitory, tangible storage media. Disk and disc, as used herein, includes compact disc (CD), laser disc, optical disc, digital versatile disc (DVD), floppy disk and Blu-ray disc, where disks usually reproduce data magnetically, while discs reproduce data optically with

lasers. Combinations of the above should also be included within the scope of computer-readable media.

Instructions may be executed by one or more processors, such as one or more digital signal processors (DSPs), general purpose microprocessors, application specific integrated circuits (ASICs), field programmable gate arrays (FPGAs), or other equivalent integrated or discrete logic circuitry. Accordingly, the terms "processor" and "processing circuity," as used herein may refer to any of the foregoing structures or any other structure suitable for implementation of the techniques described herein. In addition, in some aspects, the functionality described herein may be provided within dedicated hardware and/or software modules configured for encoding and decoding, or incorporated in a combined codec. Also, the techniques could be fully implemented in one or more circuits or logic elements.

The techniques of this disclosure may be implemented in a wide variety of devices or apparatuses, including a wireless handset, an integrated circuit (IC) or a set of ICs (e.g., a chip set). Various components, modules, or units are described in this disclosure to emphasize functional aspects of devices configured to perform the disclosed techniques, but do not necessarily require realization by different hardware units. Rather, as described above, various units may be combined in a codec hardware unit or provided by a collection of interoperative hardware units, including one or more processors as described above, in conjunction with suitable software and/or firmware.

Various examples have been described. These and other examples are within the scope of the following claims.

What is claimed is:

1. A method of decoding video data, the method comprising:

determining, based on syntax elements in a bitstream that comprises an encoded representation of the video data, residual quantized samples of a block of the video data;

determining quantized residual values based on the residual quantized samples, wherein determining the quantized residual values comprises one of:

based on the intra prediction being vertical prediction, determining the quantized residual values as:

$$Q(r_{i,j}) = \sum_{k=0}^{i} \tilde{r}_{k,j},\ 0 \le i \le (M-1),\ 0 \le j \le (N-1)$$

where $Q(r_{i,j})$ is a quantized residual value at position i,j, $\tilde{r}_{k,j}$ is one of the residual quantized samples, M is a number of rows of the block, and N is a number of columns of the block, or

based on the intra prediction being horizontal prediction, determining the quantized residual values as:

$$Q(r_{i,j}) = \sum_{k=0}^{j} \tilde{r}_{ik},\ 0 \le i \le (M-1),\ 0 \le j \le (N-1)$$

where $\tilde{r}_{i,k}$ is one of the residual quantized samples;

after determining the quantized residual values, inverse quantizing the quantized residual values;

generating predicted values by performing intra prediction for the block using unfiltered samples from above or left block boundary samples; and

reconstructing original sample values of the block based on the inverse-quantized quantized residual values and the prediction values.

2. A method of encoding video data, the method comprising:

generating a block of predicted values by performing intra prediction for a block of the video data using unfiltered samples from above or left block boundary samples;

generating residual values based on original sample values of the block and the predicted values;

quantizing the residual values;

after quantizing the residual values, determining residual quantized samples based on the quantized residual values, wherein determining the residual quantized samples comprises:

based on the intra prediction being vertical prediction, determining the residual quantized samples as:

$$\tilde{r}_{i,j} = \begin{cases} Q(r_{i,j}), & i = 0, \quad 0 \le j \le (N-1) \\ Q(r_{i,j}) - Q(r_{(i-1),j}), & 1 \le i \le (M-1), \quad 0 \le j \le (N-1) \end{cases}$$

where $\tilde{r}_{i,j}$ is a residual quantized sample at position i,j, $Q(r_{i,j})$ and $Q(r_{(i-1),j})$ are quantized residual values, M is a number of rows of the block, and N is a number of columns of the block, or

based on the intra prediction being horizontal prediction, determining the residual quantized samples as:

$$\tilde{r}_{i,j} = \begin{cases} Q(r_{i,j}), & 0 \le i \le (M-1), \quad j = 0 \\ Q(r_{i,j}) - Q(r_{i,(j-1)}), & 0 \le i \le (M-1), \quad 1 \le j \le (N-1) \end{cases}$$

$Q(r_{i,(j-1)})$ is one of the quantized residual values; and

signaling the residual quantized samples.

3. A device for decoding video data, the device comprising:

a memory configured to store the video data; and

one or more processors implemented in circuitry, the one or more processors configured to:

determine, based on syntax elements in a bitstream that comprises an encoded representation of the video data, residual quantized samples of a block of the video data;

determine quantized residual values based on the residual quantized samples,

wherein the one or more processors, are configured such that, as part of determining the quantized residual values, the one or more processors:

based on the intra prediction being vertical prediction, determine the quantized residual values as:

$$Q(r_{i,j}) = \sum_{k=0}^{i} \tilde{r}_{k,j},\ 0 \le i \le (M-1),\ 0 \le j \le (N-1)$$

where $Q(r_{i,j})$ is a quantized residual value at position i,j, $\tilde{r}_{k,j}$ is one of the residual quantized samples, M is a number of rows of the block, and N is a number of columns of the block, or

based on the intra prediction being horizontal prediction, determine the quantized residual values as:

$$Q(r_{i,j}) = \sum_{k=0}^{j} \tilde{r}_{i,k},\ 0 \le i \le (M-1),\ 0 \le j \le (N-1)$$

$\tilde{r}_{i,k}$ is one of the residual quantized samples;

after determining the quantized residual values, inverse quantize the quantized residual values;

generate predicted values by performing intra prediction for the block using unfiltered samples from above or left block boundary samples; and

reconstruct original sample values of the block based on the inverse-quantized quantized residual values and the prediction values.

4. The device of claim **3**, further comprising a display configured to display decoded video data.

5. The device of claim **3**, wherein the device comprises one or more of a camera, a computer, a mobile device, a broadcast receiver device, or a set-top box.

6. A device for encoding video data, the device comprising:

a memory configured to store the video data; and

one or more processors implemented in circuitry, the one or more processors configured to:

generate a block of predicted values by performing intra prediction for a block of the video data using unfiltered samples from above or left block boundary samples;

generate residual values based on original sample values of the block and the predicted values;

quantize the residual values;

after quantizing the residual values, determine residual quantized samples based on the quantized residual values, the one or more processors are configured such that, as part of determining the residual quantized samples, the one or more processors:

based on the intra prediction being vertical prediction, determine the residual quantized samples as:

$$\tilde{r}_{i,j} = \begin{cases} Q(r_{i,j}), & i = 0, \quad 0 \le j \le (N-1) \\ Q(r_{i,j}) - Q(r_{(i-1),j}), & 1 \le i \le (M-1), \quad 0 \le j \le (N-1) \end{cases}$$

where $\tilde{r}_{i,j}$ is a residual quantized sample at position i,j, $Q(r_{i,j})$ and $Q(r_{(i-1),j})$ are quantized residual values, M is a number of rows of the block, and N is a number of columns of the block, or

based on the intra prediction being horizontal prediction, determine the residual quantized samples as:

$$\tilde{r}_{i,j} = \begin{cases} Q(r_{i,j}), & 0 \le i \le (M-1), \quad j = 0 \\ Q(r_{i,j}) - Q(r_{i,(j-1)}), & 0 \le i \le (M-1), \quad 1 \le j \le (N-1) \end{cases}$$

where $Q(r_{(i-1),j})$ is one of the quantized residual values; and

signal the residual quantized samples.

7. The device of claim **6**, wherein the device comprises one or more of a camera, a computer, a mobile device, a broadcast receiver device, or a set-top box.

8. A device for decoding video data, the device comprising:

means for determining, based on syntax elements in a bitstream that comprises an encoded representation of the video data, residual quantized samples of a block of the video data; means for determining quantized residual values based on the residual quantized samples, wherein the means for determining the quantized residual values comprises at least one of:

means for determining, based on the intra prediction being vertical prediction, the quantized residual values as:

$$Q(r_{i,j}) = \sum_{k=0}^{i} \tilde{r}_{k,j},\ 0 \le i \le (M-1),\ 0 \le j \le (N-1)$$

where $Q(r_{i,j})$ is a quantized residual value at position i,j, $\tilde{r}_{k,j}$ is one of the residual quantized samples, M is a number of rows of the block, and N is a number of columns of the block, or

means for determining, based on the intra prediction being horizontal prediction, the quantized residual values as:

$$Q(r_{i,j}) = \sum_{k=0}^{j} \tilde{r}_{i,k},\ 0 \le i \le (M-1),\ 0 \le j \le (N-1)$$

$\tilde{r}_{i,k}$ is one of the residual quantized samples;

means for inverse quantizing, after determining the quantized residual values, the quantized residual values;

means for generating predicted values by performing intra prediction for the block using unfiltered samples from above or left block boundary samples; and

means for reconstructing original sample values of the block based on the inverse-quantized quantized residual values and the prediction values.

9. A device for encoding video data, the device comprising:

means for generating a block of predicted values by performing intra prediction for a block of the video data using unfiltered samples from above or left block boundary samples;

means for generating residual values based on original sample values of the block and the predicted values;

means for quantizing the residual values;

means for determining, after quantizing the residual values, residual quantized samples based on the quantized residual values, wherein the means for determining the residual quantized samples comprises at least one of:

means for determining, based on the intra prediction being vertical prediction, the residual quantized samples as:

$$\tilde{r}_{i,j} = \begin{cases} Q(r_{i,j}), & i = 0, \quad 0 \le j \le (N-1) \\ Q(r_{i,j}) - Q(r_{(i-1),j}), & 1 \le i \le (M-1), \quad 0 \le j \le (N-1) \end{cases}$$

where $\tilde{r}_{i,j}$ is a residual quantized sample at position i,j, $Q(r_{i,j})$ and $Q(r_{(i-1),j})$ are quantized residual values, M is a number of rows of the block, and N is a number of columns of the block, or

means for determining, based on the intra prediction being horizontal prediction, the residual quantized samples as:

$$\tilde{r}_{i,j} = \begin{cases} Q(r_{i,j}), & 0 \le i \le (M-1), \quad j = 0 \\ Q(r_{i,j}) - Q(r_{i,(j-1)}), & 0 \le i \le (M-1), \quad 1 \le j \le (N-1) \end{cases}$$

where $Q(r_{i,(j-1)})$ is one of the quantized residual values; and

means for signaling the residual quantized samples.

10. A computer-readable storage medium having stored thereon instructions that, when executed, cause one or more processors to:

determine, based on syntax elements in a bitstream that comprises an encoded representation of the video data, residual quantized samples of a block of the video data;

determine quantized residual values based on the residual quantized samples, wherein the instructions that cause the one or more processors to determine the quantized residual values include instructions that, when executed, cause the one or more processors to:

based on the intra prediction being vertical prediction, determine the quantized residual values as:

$$Q(r_{i,j}) = \sum_{k=0}^{i} \tilde{r}_{k,j}, \; 0 \le i \le (M-1), \; 0 \le j \le (N-1)$$

where $Q(r_{i,j})$ is a quantized residual value at position i,j, $\tilde{r}_{k,j}$ is one of the residual quantized samples, M is a number of rows of the block, and N is a number of columns of the block, or

based on the intra prediction being horizontal prediction, determine the quantized residual values as:

$$Q(r_{i,j}) = \sum_{k=0}^{j} \tilde{r}_{ik}, \; 0 \le i \le (M-1), \; 0 \le j \le (N-1)$$

where $\tilde{r}_{i,k}$ is one of the residual quantized samples;

after determining the quantized residual values, inverse quantize the quantized residual values;

generate predicted values by performing intra prediction for the block using unfiltered samples from above or left block boundary samples; and

reconstruct original sample values of the block based on the inverse-quantized quantized residual values and the prediction values.

11. A computer-readable storage medium having stored thereon instructions that, when executed, cause one or more processors to:

generate a block of predicted values by performing intra prediction for a block of the video data using unfiltered samples from above or left block boundary samples;

generate residual values based on original sample values of the block and the predicted values;

quantize the residual values;

after quantizing the residual values, determine residual quantized samples based on the quantized residual values, wherein the instructions that cause the one or more processors to determine the residual quantized samples comprise instructions that, when executed, cause the one or more processors to:

based on the intra prediction being vertical prediction, determine the residual quantized samples as:

$$\tilde{r}_{i,j} = \begin{cases} Q(r_{i,j}), & i = 0, \; 0 \le j \le (N-1) \\ Q(r_{i,j}) - Q(r_{i,(j-1)}), & 1 \le i \le (M-1), \; 0 \le j \le (N-1) \end{cases}$$

where $\tilde{r}_{i,j}$ is a residual quantized sample at position i,j, $Q(r_{i,j})$ and $Q(r_{(i-1),j})$ are quantized residual values, M is a number of rows of the block, and N is a number of columns of the block, or

based on the intra prediction being horizontal prediction, determine the residual quantized samples as:

$$\tilde{r}_{i,j} = \begin{cases} Q(r_{i,j}), & 0 \le j \le (N-1), \; j = 0 \\ Q(r_{i,j}) - Q(r_{i,(j-1)}), & 0 \le i \le (M-1), \; 1 \le j \le (N-1) \end{cases}$$

where $Q(r_{i,(j-1)})$ is one of the quantized residual values; and

signal the residual quantized samples.

* * * * *